REVUE MENSUELLE N° 5 - NOVEMBRE 85 - PRIX 20 F

exelement vôtre



- la synthèse de la parole
- banc d'essai "exeldrums"
- programmes de recopie d'écran
- listings des gagnants du concours

sommaire

Novembre 1985
REVUE MENSUELLE
Prix: 20 F

REDACTION: Patrice Chaillan

ONT COLLABORE: à la conception de ce numéro, toute l'équipe d'EXELVISION de SOPHIA.

ILLUSTRATION: Catherine Chaillan

PHOTOS: Massé

DIRECTEUR de la PUBLICATION: Exelvision

RELATIONS LECTEURS: Exelvision - Courrier Exelment Vôtre - Place Joseph Bermond Immeuble Ophira - 06560 Valbonne

ABONNEMENTS: 6 numéros 100 F au lieu de 120 F, frais de port compris. Règlement à l'ordre d'Exelvision.

PUBLICITE: Ecrire au journal et envoyer textes ou typons au format, au plus tard 3 semaines avant parution.

DIFFUSION: points de vente Exelvision.

COUVERTURE:
HUGUES / MASSE

Imprimé par
l'Atelier d'Impression

la reconnaissance et la synthèse de la parole

1.1 Situation actuelle

Un ordinateur qui parle et qui comprend des ordres parlés, voilà le pari des scientifiques pour la décennie à venir. Mais avant de plonger dans une science qui n'est pas vraiment fiction, nous pouvons examiner l'état du developpement de la synthèse et la reconnaissance de la parole à l'heure actuelle.

La synthèse vocale constitue un immense marché d'avenir.

Depuis les années 50, des chercheurs de différents pays se sont penchés sur la synthèse vocale. Recréer la voix humaine à partir de micro-processeurs et de logiciels a necessité d'importants investissements tant en facteur pécuniaire qu'en facteur temps. L'aboutissement de ces recherches s'est concrétisé dans la distribution grand public par la mise en vente sur le marché de différents produits tels que:

Les jeux éducatifs de Texas Instrument: Speak and Spell

-Appareillage des automobiles Renault et autres
-Standardisation de la synthèse vocale pour différents objets de consommation: Pèse personnes, alarme... etc...

1.2 L'avenir

Le secteur professionnel a été touché par la vague des ordinateurs parlants. Ainsi de nombreux terminaux à utilité publique, terminaux bancaires, point d'informations etc... sont maintenant équipés de circuits de synthèse vocale.



Les conditions de developpement de la synthèse vocale:

Les facteurs déterminant le succès de la synthèse vocale sont:

1) Le prix de revient d'un circuit
2) L'implantation du circuit dans un appareil donné

Le prix de revient des circuits de synthèse vocale ainsi que leurs prix d'interfaçage avec un appareil quelconque tendent à diminuer. Néanmoins, ces prix restent encore elevés freinant ainsi une distribution massive.

Le tableau (voir page 3) fait ressortir les perspectives de marché pour les années à venir.

Un fait important à signaler: La bonne tenue sur le marché des produits éducatifs de Texas Instruments s'explique par le public ciblé dans le marketing. Une clientèle de moins de dix ans qui s'accomode d'une voix de type robot. Il est certain qu'un public plus agé exige une meilleure qualité de prononciation.

La synthèse vocale demande aussi une place mémoire assez importante. La phonétisation d'une lettre de l'alphabet occupe environ 100 octets. Un mot usuel occupe donc environ 300 octets. Le prix du Ko RAM étant à la baisse, le problème de la taille mémoire ne devrait plus constituer un frein à la distribution. Cette place mémoire peut être diminuée grâce au système de prédiction linéaire qui permet d'échantilloner sur 25 millisecondes la prosodie l'energie et la fréquence d'un son, une seconde de "parole" occupant avec ce système 1200 bits de mémoire.



Synthèse de la parole:
Marché et perspectives

Type d'applications	Domaines d'applications	Date de début	Date de généralisation	Marché mondial en millions $	Prix acceptable
Retour vocal Diffusion informations Alarmes	Automatismes industriels	1980	1983		
Terminaux intelligents	Terminaux informatiques	1985	1990	185	< 500
Banques de données	Télématique	1976	1985	224	
Automobiles	Grand public	1981	1985		50

La prolifération sur le marché de la micro-informatique de cartes traitant la synthèse de la parole laisse entrevoir aux constructeurs l'engouement des utilisateurs de micro pour la synthèse vocale.

-Smoothtalker de Mac Intosh.
-La carte Matra pour Apple IIe
-Synthétiseur pour Oric Atmos
-Thomson est son synthétiseur de parole pour la gamme MO5 et TO7/70
-Echo cricket (TMS 5220) d'Apple
-Carte Ferm pour Apple

Les utilisateurs potentiels de la synthèse vocale sont notemment:

-Les non voyants par l'intermédiaire de leur association VALENTIN HAUY
-Les personnes handicapées au niveau de la motricité de la parole
-L'éducation

Les autres applications de la synthèse vocale sont à rechercher dans le domaine public.

Les banques de données se dotent de synthèse vocale pour guider un utilisateur dans ces recherches. Certaines sociétés voient dans la synthèse vocale une application au niveau de la prise des commandes en VPC. La Redoute, les 3 Suisses commencent à équiper les ordinateurs de circuits synthétisant la parole.

Les banques et les grands organismes publics considèrent la synthèse vocale non pas comme un gadget, mais comme une necessité afin de rendre leurs systèmes informatiques plus conviviaux. La parole constituant l'acte le plus naturel pour se faire comprendre.Le secteur tertiaire utilisera massivement la synthèse vocale. devront dans l'avenir guider la personne vocalement.

1.3 Là où les choses se compliquent

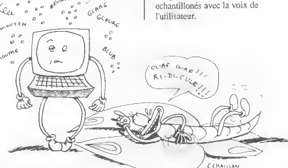
Si le fait de faire parler un ordinateur était relativemant facile, la reconnaissance de la parole consitue un véritable exploit. Nous allons éxaminer les différents freins au developpement de la reconnaissance de la parole.

Les systèmes monolocuteurs et multilocuteurs.

L'ordinateur est une machine, et en tant que tel, il ne fait que comparer des données. Or la comparaison des données dans la reconnaissance vocale est le point épineux. Pour comparer des données il faut:

-a. Que des mots de référence soient intégrés à l'ordinateur

-b. que ces mêmes mots soient echantillonés avec la voix de l'utilisateur.



La deuxième condition implique qu'il est beacoup plus facile de développer un système monolocuteur que multilocuteur.Unsystème monolocuteur est un système ne faisant intervenir qu'un locuteur, soit un seul utilisateur potentiel de l'ordinateur.

Un autre frein au developpement:

-a. L'analyse physique de la voix

La voix change de timbre suivant l'heure, la tension, les ennuis pathologiques (rhumes, angine etc..). Il est très difficile d'optimaliser l'analyse physique d'une voix. Cette analyse nécessite une énorme capacité mémoire utilisateur.

-b. L'analyse sémantique

Si la raconnaissance vocale est fiable sur une quantité x de mots, la compréhension d'une phrase est extraordinairement complexe à traiter. L' analyse sémantique est le problème majeur de la reconnaissance vocale. En effet les cas de figures dans la compréhension d'une phrase sont très difficile à quantifier de même que l'intonnation d'une phrase qui peut radicalement changer le sens de celle-ci.

1.4 Un gigantesque marché

Même si des problèmes se posent, la reconnaissance de la parole est le facteur clé pour l'implantation durable des systèmes informatiques. Tous les secteurs de l'activité humaine seront touchés:

-Transport
-Medecine
-Tertiaire
-Militaire
-Industriel
-ordinateur personnel

1.5 Et Exelvision dans tout ce dédale

Chers lecteurs et lectrices, savez vous qu'Exelvision est le seul constructeur à proposer un ordinateur avec synthèse vocale intégrée grâce au processeur TMS 5220A. La collaboration avec le staff de Texas Instruments a fait que votre ordinateur pronnonce des mots correctement à l'opposé de certains ordinateurs qui emettent des sons étranges.

Alphabet phonétique	Terminologie	Exemples
/I/	i	*li* t
/e/	é fermé	été
/u/	ou	*lou*p
/j/	semi-voyelle	*y*eux
/p/	Consonnes	*p*an

Exemples de quelques phonèmes

Concrètement, si vous possédez une CRAM, des cassettes contenant des données speech sont disponibles. Des effets sonores aux mots les plus usuels, ces cassettes sont indispensables si vous désirez utiliser le synthetiseur vocal de votre EXL 100.

2.6 Des précisions plus techniques

Nous allons aborder dans ce paragraphe, les aspects techniques liés à la synthèse de la parole. Ces éléments techniques nous ont été communiqués par Alain Marafetti.

2.6.1 Les phonèmes

Définition: Un phonème représente la plus petite unité phonétique d'un langage

A partir de ces phonèmes, il est théoriquement possible de reconstituer tous les mots d'un vocabulaire donné. L'assemblage des phonèmes entre eux afin de former un mot suit des lois particulières. Les phonéticiens (personnes spécialisées dans la phonétique ou la prononciation d'un mot)) ont dénombré 37 phonèmes dans la langue française comprenant 16 voyelles, 18 consonnes et 3 semi voyelles.

Cependant l'association de plusieurs phonèmes ne donne pas nécessairement un mot compréhensible dans un langage parlé. Ainsi "p.." et " an " ne donne pas forcemment PAN. La juxtaposition des deux phonèmes conduit seulement à la perception de deux sons différents séparés par un temps mort. C'est la transition d'unphonème vers un second phonème qui rend le mot intelligible.

Les diphonèmes ou phonatome

Les diphonèmes désigne l'ensemble formé par l'association de deux phonèmes. Une analyse statistique réalisée au LIMSI à Orsay a montré qu'à partir d'un répertoire de 627 phonèmes, il était possible de reconstituer n'importe quelle phrase en français.

Les différents types de synthétiseurs

Schématiquement, il existe deux types majeurs de synthétiseurs capables de réaliser une synthèse sonore.

-a Les synthéthiseurs à FORMANTS

-b Les synthétiseurs à prédiction linéaire
C'est le dernier type de synthétiseurs qui est utilisé par EXELVISION et IPSO synthétiseurs, nous laisssons le soin au lecteur d'approfondir son étude en consultant l'excellent ouvrage deMarc Ferreti et François Cinare - Synthèse, reconnaissance de la parole- aux éditions *édi* **TESTS**.

Difficultés d'adaptatinn sur une maacine

1) La taille du vocabulaire

Le problème qui se pose est le stockage du vocabulaire dans la machine.

-Vocabulaire illimité: Cette machine serait idéale pour les non voyants et les centres de renseignement automatiques. Le vocabulaire pour des raisons de taille mémoire fait intervenir une synthèse à partir du texte à dire.

-Vocabulaire limité: Quelques mots sont suffisants: Cas des alarmes. Le vocabulaire est fini est ne fait pas intervenir de synthèse à proprement parler.

-Vocabulaire limité mais modifiable par l'utilisateur: Ce type de solution semble être celui adopté par les constructeurs d'ordinateurs.

2) Qualité de la voix

La qualité sonore de la voix est un élément essentiel dans la synthèse de la parole . Pour une application publique une voix féminine est obligatoire. Par contre au niveau d'applications professionnelles une voix féminine serait mal perçue. Les échantillonages de voix font l'objet de tests impitoyables. En effet la voix qui va être synthétisée doit avoir une bonne qualité en intelligibilité et en agrément.

3) Le coût du produit

Pour unr application grand public le prix du produit est un facteur primordial à l'opposé d'un produit ellitiste (ordinateur de bord d'avions ou autres automatismes industriels. (Voir tableau)

Etat actuel du developpement

On peut distinguer plusieurs modes de synthèse vocale:

- La synthèse par mots permettant avec une qualité optimale d'écouter un mot , celui-ci étant digitalisé dans son entier. Ce mode de synthèse est malheureusement limité par la bibliothèque de mots enregistrés. (numération parlée, ennoncé des nombres) ; Par exemple Mille neuf cent utilise les nombres mille neuf et cent.

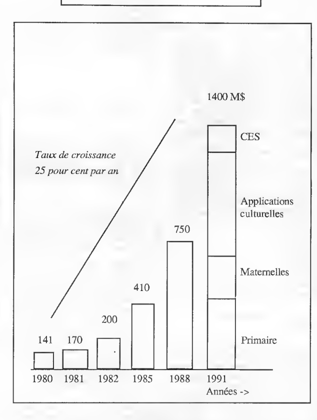
-La synthèse par éléments phonétiques

-Les syllabes : Ce procédé permet de crééer des mots à condition que les lois d'assemblage de ces syllabes ne soient pas trop complexes. (pour dire automatique on emploiera les syllabes "autio" , "mati" , "que"

-Les phonèmes : Ce procédé d'assemblage des plus petites unités phonétiques permet de reconstituer les mots parlés. Les règles d'assemblage sont très complexes dans ce procédé et on lui préfère le procédé des diphonèmes.

-Les diphonèmes: Une liste de 1200 diphonèmes permet de synthétiser n'importe quel message de la langue française, certains système tel l'icophone V du LIMSI n'utilisant que 627 diphonèmes. Dans ce système les règles d'assemblage sont relativement simples, le traitement principal consistant à commander la prosodie de la phrase, (la prosodie étant l'intonation et le rythme d'un mot ou d'une phrase).

Références bibliographiques
-sybthèse et reconnaissance de la parole
Editions *édi* **TEST**
-Revue sciences et techniques
Numéro 2 Hors série



Evolution du marché mondial des jeux éducatifs parlants

exeldrums

Ce mois-ci, nous avons passé au test le logiciel Drums plus son interface.

2.1 Présentation

Livré avec son extension "hardware" ce logiciel regroupe toutes les fonctions disponibles et plus encore sur les boites à rythmes modernes. Pour ceux qui ne connaissent pas les boites à rythmes, nous allons expliquer leurs modes de fonctionnement.

Depuis l'apparition de micro-processeurs, les musiciens disposent de certains outils qui facilitent considérablement leur travail tant en composition que dans les séances d'enregistrement. En effet les synthétiseurs font partie de la panaplie disponible pour les musiciens amateurs ou professionnels.

Une boite à rythme n'est plus ni moins qu'un synthétiseur de son jouant dans des fréquences correspondant aux divers instruments d'une batterie.

De plus, une boite à rythme tiendra toujours le tempo, ce qui n'est pas évident même pour certains batteurs.

2.2 codification de l'information

Une boite à rythme suit toujours le même mode de programmation.

L'unité minimale sera la double croche. Bien qu'il ne soit pas nécessaire de connaître le solfège pour utiliser EXELDRUMS, il est quand même avantageux d'avoir quelques notions de solfège rithmique.

Nous allons pour des questions de simplicité prendre des exemples de mesure binaire simple. Le 4/4 constitue le meilleur exemple.



a. La mesure:

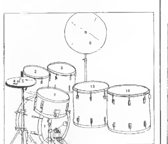
Un morceau de musique est divisé en mesure. Une mesure en 4/4 implique que chaque mesure comprendra 4 temps et que l'unité rythmique de la mesure vaudra un temps soit une noire.

Notre unité minale de programmation sur EXELDRUMS est la double croche. Il faut quatre double croche pour faire une noire soit un temps. Comme notre mode de programmation est la double croche, nous aurons 16 pas pour représenter une mesure de 4/4

CORRESPONDANCE ENTRE LES NOTES ET LA GRILLE (FRAME) DE LA BOITE A RYTHME

En tête de la grille

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
RONDE	■															
BLANCHE	■								■							
NOIRE	■				■				■				■			
CROCHE	■		■		■		■		■		■		■		■	
DBLE CROCHE	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■



2.3 utilisation d'EXELDRUMS

Nous pouvons dire que DRUMS est excessivement facile d'emploi par rapport à d'autres boites à rythme qui demande une bonne connaissance du solfège

a. Composition d'une mesure

L'écran de votre ordinateur affiche un tableau avec un curseur qui défile plus ou moins vite en fonction du tempo choisi. Pour programmer un rythme, il suffit de déplacer le curseur et placer les "beat" ou battements sur une ligne représentant un intrusment.

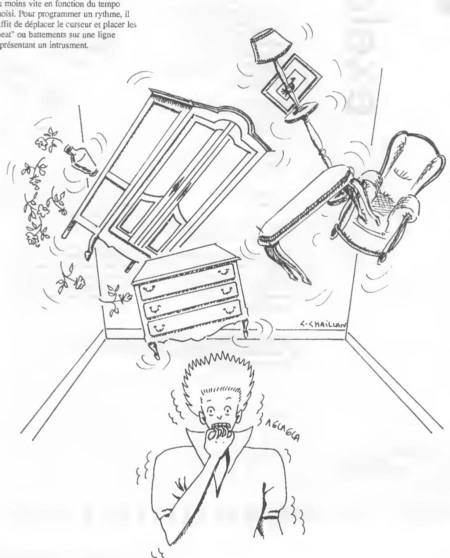
2.3.1 Choix des instruments

-16 instruments et un roulement de caisse claire sont disponibles. (Voir tableau 2.1).

Le fait de visualiser à l'écran les pas permet de programmer cette boite à rythme très facilement.

Le tableau 2.2 montre la correspondance entre les différentes valeurs de notes et leur programmation sur EXELDRUMS.

Si vous voulez jouer une ronde soit 4 temps dans une mesure 4/4, il suffit de valider une fois le curseur. La ronde est jouée tous les 16 pas.



b. Composition d'un morceau de musique

Maintenant que vous avez programmé une mesure, vous désirez composer un morceau qui comprend plusieurs mesures et si possible des mesures qui ne se ressemblent pas. Et oui nous pouvons le dire, DRUMS fait la différence grâce à la manière d'enchaîner les différentes mesures. En effet un tableau est prévu pour composer un morceau.

Ce tableau affiche les mesures que vous avez composé. Supposons que vous ayez programmé quatre mesures. Celles-ci seront notées dans votre tableau en fonction de leur ordre de composition 1, 2, 3, 4. A coté de chaque mesure un emplacement est prévu pour indiquer le nombre de répétitions de la mesure.

Exemple :
Vous pouvez faire jouer 5 fois la première mesure, 4 fois la seconde, 1 fois la troisième et quatre fois la quatrième et ensuite revenir au début etc...



2.4 Sauvegarde et chargement des morceaux

EXELDRUMS vous permet de sauvegarder et recharger vos modèles de rythmes ainsi que vos morceaux.

La sauvegarde et le chargement peuvent se faire avec:

-a. Un magnétophone à cassettes
-b. une exelmémoire

2.5 Utilisation à partir d'un programme BASIC

Si vous disposez d'EXELDRUMS, vous pouvez utilisez un programme BASIC pour faire jouer la boite à rythme. Ce scenario peut se présenter si pour une raison quelconque vous désirez inclure des rythmes dans un programme.

2.6 Connexion à un amplificateur stéréophonique

EXELDRUMS offre la possibilité à l'utilisateur de se connecter directement à l'entrée d'un amplificateur ou d'une table de mixage. Alors là, bonjour l'enfer pour les voisins, ça deménage un maximun. Le son est très bon, et on peut utiliser EXELDRUMS dans des applications professionnelles.

2.7 Rythmes préprogrammés

EXELDRUMS possède 18 rytmes préprogrammés qui vous rendront bien des services. Cha cha, tango, valse, rock binaire ou ternaire, break dance, disco etc...

2.8 Conclusion

Proposé avec son extension interface à un prix voisin de 1000 francs, ce logiciel tranforme votre EXL 100 en batteur professionnel. La qualité du son est meilleure que bien des boites à rythmes qui coûtent facilement du double au quintuple. Grâce à son circuit logique developpé par HOHNER le son permet une reproduction parfaite dans tous les registres. Si vous êtes musicien amateur ou professionnel, n'hésitez pas et dotez vous d'un support rythmique indispensable à toute réalisation musicale.

les boucles automatiques

3.1 Les boucles automatiques FOR ...NEXT

Nous allons aborder dans cette rubrique la notion de boucle automatique. La notion de boucle est un concept qui s'utilise très souvent dans le langage de programmation. Lorsque vous pensez à une boucle, vous visualisez une figure géométrique ressemblant à un cercle.

La meilleure façon d'expliquer le concept de boucle est de taper un programme. Cf listing 3.1

Commentaires:

La ligne 150 initialise la variable A (A vaut zéro)
La ligne 160 affiche A à l'écran
La ligne 170 incrémente la valeur de A (A=A+1)
La ligne 180 renvoie le programme à la ligne 160

Le programme tournera perpétuellement. En effet ce programme ne pourra être interrompu qu'avec une instruction <CTL C>. Si vous suivez bien le cheminiement du programme, vous vous apercevrez que le programme forme une boucle.

160
165
170
180

Nous allons modifier ce petit programme en introduisant à la ligne 175 un test. Cette ligne testera la valeur de A afin de stopper le programme lorsque A atteindra une certaine valeur.

175 IF A=12 THEN GOTO 200

200 CLS:LOCATE(18,7):PRINT "< A VAUT MAINTENANT 12">

Lorsque A vaudra 12 le programme affichera les instructions de la ligne 200 et s'arrêtera.

La variable A a pris successivement les valeurs
1,2,3,4,5,6,7,8,9,10,11,12

Il existe deux instructions BASIC qui permettent d'obtenir le même résultat. Ces instructions sont respectivement:

FOR et NEXT

3.2 Format et syntaxe

FOR [variable=valeur1] TO [valeur2]

Corps de la boucle

NEXT [variable]

Avant d'examiner ces instructions de plus près, tapez le programme (Cf listing 3.3).



Commentaires:

La ligne 150 déclare un début de boucle:
FOR I=1 TO 12. Il faut comprendre pour I variant de 1 à 12.
A prendra des valeurs croissantes de 1 à 12.
I s'appellera la variable de boucle
Les valeurs que prendra I sont croissantes
La ligne 160 : PRINT I, permet d'afficher la variable de boucle
La ligne 170 NEXT I: Comprendre "prochaine valeur de I", permet d'affecter à I la prochaine valeur.

Le déroulement de ce mini programme est le suivant:
Ligne 150 : I prend la première valeur spécifiée soit 1
Ligne 160 : Le programme affiche I soit 1
Ligne 170 : I est incrémenté et prend comme valeur I+1 soit 2
Le programme revient à la ligne 150 jusqu'à ce que I atteigne la valeur 12.

Ce programme est équivalent au programme 3.4.

Commentaires:

Ligne 150: initialisation de I à la valeur 1
Ligne 160: affichage
Ligne 170: test de la variable I
Ligne 180: Incrémentation de I
Ligne 190: saut à la ligne 160

L'utilisation d'une instruction FOR...NEXT nous fait gagner 3 lignes d'instructions soit environ 80 octets.

Cette instruction peut se rajouter à la suite d'une déclaration de début de boucle. STEP signifie le pas de la boucle, c'est à dire la valeur d'incrémentation de la variable de boucle. Lorsque l'option STEP n'est pas spécifiée, le pas est pris égal à 1 par défaut.

Cette option est très utile car elle permet d'utiliser les boucles en valeurs décroissantes, en croissance exponentielle, etc...

Exemple: Boucle de pas négatif

Cf listing 3.5



Commentaires:

Nous avons simplement déclaré une instruction de boucle avec comme valeur initiale de I, la valeur 12 et comme valeur finale, la valeur 1. Le pas est de -1.

Toutes sortes de pas peuvent être fournis, mais il faut faire attention à ce que le pas ne soit pas incompatible avec les valeurs initiales et finales. Dans notre exemple, un pas de -13 génèrerait une erreur.



Un bon exemple de l'utilisation d'une boucle!
Cf listing 3.6

3.4 Champs d'applications des boucles

Les boucles sont généralement utilisées dans toutes sortes d'applications. Le plus simple pour nous consiste à partir d'un programme simple de gestion que nous allons analyser.

Une entreprise vous demande d'établir un programme de paye. La société emploie 12 personnes. Le programme devra prévoir les retenues salariales, ainsi que la gestion des heures chômées.

Pour établir un bulletin de paye d'un salarié "x", le programme devra demander à l'utilisateur:

-a. Le nom de la personne
-b.Sa qualification professionnelle
-c. Le nombre d'heures de présence

Cette partie constitue la saisie des données. La partie calcul ne pose pas de problèmes et sera traitée au cours du programme.

Les boucles vont être utilisées dans ce programme. En effet il serait stupide d'écrire 12 instructions pour demander le nom des salariés. Il est donc recommandé de passer par une boucle qui variera de 1 à 12, soit le nombre des employés.

3.5 Introduction de données à l'intérieur d'une boucle

La première précaution à prendre est de dimensionner un tableau si votre variable peut prendre plus de 10 valeurs. Dans le cas qui nous occupe, il faudra déclarer un tableau, car nous devons traiter 12 personnes.

**L'introduction des données doit se faire dans le corps de la boucle. En effet si vous voulez répeter 12 fois l'introduction du nom, du prénom, etc.. Il faut impérativement que l'instruction INPUT soit écrite dans le corps de la boucle.*

Le listing 3.7 permet de réaliser une introduction de 12 noms.

Commentaires:
Ce programme comprend deux parties:

-1. Ecriture des noms dans la table des noms (N$): Ligne 170

-2. Lecture des noms dans la table (N$): Ligne 220

La ligne 170 permet d'introduire des noms dans la table. La variable N$ est indicée avec la variable de boucle I.

Ceci s'écrit N$(I). On va écrire successivement 12 valeurs dans la table des noms. Si nous donnons aux différents tours de boucle les valeurs A ,B, C, D, E, F, G, H, I,J , K, L, nous obtiendrons la table suivante:

Tour	variable	valeur
1	N$(1)	A
2	N$(2)	B
3	N$(3)	C
....	....	
....	....	
12	N$(12)	L

L'indice ou la variable de boucle est croissant dans ce programme. Grâce à cette organisation en tableau, nous pouvons écrire ou lire un nom en fournissant son indice.

Si nous voulons lire le troisième élément de la table, vous pouvez taper en mode direct ou programmé, PRINT N$(3), ce qui donnera **"C"**.

De par la structure de table, les recherches au sein de celle-ci se feront facilement. Mais revenons à notre programme de gestion. Nous voulons saisir les noms, les prénoms, la qualification, le taux horaire, le nombre d'heures effectué dans le mois. Pour cela, nous écrirons dans le corps de notre boucle, tout comme pour les noms:

INPUT "Prénom:";P$(I) etc...

De même, à la lecture de la table nous afficherons les différentes rubriques. Nous allons compléter notre programme de saisie, cf listing 3.8.

Commentaires:

Lignes 170 à 350 : Ecriture de la table avec calculs et tests sur les différentes entrées.

Lignes 360 à 580 : Lecture et affichage de la table

*Nous avons introduit toutes les valeurs dans des variables alphanumériques. Lorsque nous devons traiter ces chaînes de caractères en calcul, nous utilisons la fonction VAL (N$(I)).

Le programme est somme toute assez simple. La seule gène réside dans la manipulation des indices et des niveaux de parenthèse. Un excellent exercice de "style" consisterait dans la refonte des tables en un seul tableau en vous aidant de EV5.

3.6 Les boucles imbriquées

Il est très fréquent dans un programme de rencontrer des boucles à l'intérieur d'une boucle. Cela ne pose aucun problème de programmation. Il faut seulement faire attention à l'ordre de sortie des boucles.

Exemple: 2 boucles imbriquées

```
FOR I= a TO b

FOR J= c TO d
        .
        .
        .
        .
NEXT J

NEXT I
```

Exemple: Les tables de multiplication Cf Listing 3.9

Commentaires:

Faire attention à l'ordre de sorties des boucles.

3.7 Utilisation d'une variable de cumul

Lorsque vous utilisez des boucles pour écrire une table, il est parfois nécessaire d'utiliser une variable de cumul pour stocker des résultats. Le problème peut se poser si, par exemple, vous voulez calculer les dépenses d'un mois. Nous allons utiliser une variable de cumul qui stockera toutes les sommes dépensées (Cf listing 3.10)

Commentaires:

Ligne 150 : Demension des tables
Ligne 160 : Initialisation de la variable de cumul
Ligne 170 : Début de la boucle
Lignes 190 à 220: Saisie du nom de l'article et de son prix et test sur p$(I)
Ligne 270 à 320 : Lecture des tables et écriture du total

En fait, l'utilisation des variables de cumul est très simple, et on peut décrire son mode d'utilisation de cette manière:

-a. Avant une boucle, on initialise la variable de cumul à 0. Ceci implique que cette variable vaudra 0 au commencement de la boucle.

-b. Immédiatement après le calcul que nous désirons stocker, il faut écrire:

VARIABLE CUMUL=VARIABLE CUMUL + Var (I)

Ceci a pour effet d'incrémenter la variable de cumul d'une certaine valeur à chaque tour de boucle.

Dans notre exemple, la variable de cumul est égale à TOTAL, le prix est contenu dans P$(I).

Conclusion:

L'utilisation des boucles automatiques dans un programme ne comporte aucune difficulté. Pratiquement tous les programmes utilisent des boucles et des tables. Les jeux d'aventure et les jeux de rôle comportent beaucoup de boucles et de tables, et c'est avec un grand plaisir que je vous annonce que le numéro de JANVIER sera un numéro spécial jeux d'aventure et de rôle. La revue sera consacrée dans son entier à la réalisation d'un jeu d'aventure.

```
100 '################################
110 '#            LISTING 3.1        #
120 '################################
130 CLS "WbC"
140 CALL COLOR("OWb")
150 A=0
160 LOCATE (5,5):PRINT "A VAUT- - ",A
170 A=A+1
180 GOTO 160
```

```
100 '################################
110 '#            LISTING 3.1        #
120 '################################
130 CLS "WbC"
140 CALL COLOR("OWb")
150 A=0
160 LOCATE (5,5):PRINT "A VAUT--->";A
170 A=A+1
175 IF A>=12 THEN GOTO 200
180 GOTO 160
200 LOCATE (5,5):PRINT "A VAUT- ,";A
205 LOCATE (18 7):PRINT "A VAUT MAINTENANT,";A
```

```
100 '################################
110 '#            LISTING 3.1        #
120 '################################
130 CLS "WbC"
140 CALL COLOR("OWb")
150 FOR I=1 TO 12
160 PRINT I
170 NEXT I
```

```
100 '################################
110 '#            LISTING 3.1        #
120 '################################
130 CLS "WbC"
140 CALL COLOR("OWb")
150 A=1
160 IF A>12 THEN END
170 PRINT A
180 A=A+1
190 GOTO 160
```

```
100 '################################
110 '#            LI TING  ,1        #
120 '################################
130 CLS "WbC
140 CALL COLOR("OWb")
150 FOR I=1, TO 1 STEP  1
160 PRINT I
170 NEXT I
```

```
100 '################################
110 '#            LISTING  ,1        #
120 '################################
130 CLS "WbC"
140 CALL COLOR("OWb")
150 PRINT " -- -   TRACE DE Y=SIN(X   - -
160 CALL HRON "E" 5 14
170 FOR I=0 TO 2*PI STEP PI 45
180 Y=SIN(I)*60
190 CALL PLOT("W" I*50 Y+70)
200 NEXT
```

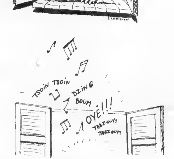
```
100 '################################
110 '#            LI TING   I        #
120 '################################
130 CLS "WbC"
140 CALL COLOR  OWb")
150 DIM N$ 12)
160 FOR I 1 TO 12
170 LOCATE (5 5):INPUT  NOM   N$(I)
180 NEXT I
190 'LECTURE DE LA TABLE
200 FOR I=1 TO 12
210 CLS
220 LOCATE (5,5):PRINT  NOM:",N$(I)
230 GOSUB 260
240 NEXT I
250 END
260 'SOUS PROGRAMME PROCHAIN NOM
270 LOCATE (18 5) PRINT
    "(APPUYEZ SUR [ENVOI] POUR SUITE)
280 PAUSE
290 RETURN
```

```
100 '####################################
110 '#            LISTING 3.1           #
120 '####################################
130 CLS "WBC"
140 CALL COLOR("OWB")
145 '##########DIM TABLE################
150 DIM N$(12) F$(12) Q$(12) NH$(12) T$(12)
    SALAIREBRUT(12)
160 DIM SECU(12) VIEL(12) RETE(12) SALAIRENET(12)
165 '##########ECRITURE TABLE############
170 FOR I=1 TO 12
180 CLS
190 LOCATE (5 5):INPUT "NOM:";N$(I)
200 LOCATE (6 5):INPUT "PRENOM:";F$(I)
210 LOCATE (7 5):INPUT "QUALIFICAITON:";Q$(I)
220 IF VAL(Q$(I))=1 OR VAL(Q$(I))=2 OR VAL(Q$(I))=3
    THEN 230 ELSE 210
230 LOCATE (8 5):INPUT "NOMBRE D'HEURES:";NH$(I)
240 IF NOT NUMERIC(NH$(I))THEN 230
250 LOCATE (9 5):INPUT "TAUX HORAIRE:";T$(I)
260 IF NOT NUMERIC(T$(I))THEN 250
290 '#####CALCUL DU SALAIRE NET##########
300 SALAIREBRUT(I)=VAL(T$(I))*VAL(NH$(I))
310 SECU(I)=SALAIREBRUT(I)*.08
320 VIEL(I)=SALAIREBRUT(I)*.025
330 RETE(I)=SECU(I)+VIEL(I)
340 SALAIRENET(I)=SALAIREBRUT(I)-RETE(I)
350 NEXT I
360 '########LECTURE DE LA TABLE#########
380 FOR I=1 TO 12
381 CLS
391 LOCATE (3 5):PRINT "BULLETIN DE PAYE"
400 LOCATE (5,5):PRINT "NOM                :";N$(I)
410 LOCATE (6 5):PRINT "PRENOM             :";F$(I)
420 LOCATE (7 5):PRINT "QUALIFICATION      :";Q$(I)
430 LOCATE (8 5):PRINT "NOMBRE D HEURES:";N$(I)
440 LOCATE (9,5):PRINT "TAUX HORAIRE       :";T$(I)
450 LOCATE (10 5):PRINT "---------------------------"
460 LOCATE (11 5):PRINT "SALAIRE BRUT  :"
470 PRINT USING"######.##";SALAIREBRUT(I)
480 LOCATE (13 5):PRINT "RETENUE CPAM  :";
490 PRINT USING"######.##",SECU(I)
500 LOCATE (14 5):PRINT "RETENUE VIEIL :";
510 PRINT USING"######.##";VIEL(I)
520 LOCATE (15 5):PRINT "---------------------------"
530 LOCATE (16,5):PRINT "TOTAL RETENUE :";
540 PRINT USING"######.##";RETE(I)
541 LOCATE (18,5):PRINT "---------------------------"
550 LOCATE (19 5):PRINT "SALAIRE NET   :";
560 PRINT USING"######.##";SALAIRENET(I)
561 LOCATE (20 5):PRINT "
570 GOSUB 600
580 NEXT I
590 END
600 'SOUS PROGRAMME PROCHAIN NOM
610 LOCATE (22 5):PRINT "'APPUYEZ SUR [ENVOI]
    POUR SUITE "
620 PAUSE
630 RETURN
```

```
100 '####################################
110 '#            LISTING 3.9           #
120 '####################################
130 CLS "WBC"
140 CALL COLOR("OWB")
150 FOR I=1 TO 9
160 FOR J=1 TO 9
170 R=I*J
180 LOCATE (3,15):PRINT "TABLE DE ";I
190 LOCATE (5+J,15):PRINT I;"X";J;"=";R
200 PAUSE .2
210 NEXT J
220 CLS
230 NEXT I
```

```
100 '####################################
110 '#            LISTING 3.9           #
120 '####################################
130 CLS "WBC"
140 CALL COLOR("OWB")
150 DIM N$(31) F$(31)
160 TOTAL=0
170 FOR I=1 TO 31
180 CLS
190 LOCATE (5 5):INPUT "NOM DE L'ACHAT:";N$(I)
200 IF N$(I)=""THEN 250
210 LOCATE (6,5):INPUT "PRIX              :";F$(I)
220 IF NOT NUMERIC(F$(I))THEN 210
230 TOTAL=TOTAL+VAL(F$(I))
240 NEXT I
250 'AFFICHAGE
260 CLS
270 LOCATE (5,5)
280 FOR I=1 TO 31
290 IF N$(I)=""THEN 320
300 PRINT TAB(5)N$(I);"            ";F$(I)
310 NEXT I
320 PRINT:PRINT:PRINT TAB(5)"TOTAL=";TOTAL
```





les sous-programmes

4.1 GOSUB et RETURN:

Définition:

Il est fréquent qu'une séquence d'instructions soit utilisée **plusieurs fois** dans un même programme.A ce sujet, et pour illustrer notre propos, nous avons proposé dans Exelement votre n°2 un programme nommé EXLASTRE.

Ce programme comprenait des séquences d'instructions identiques, et cela par 12 fois.Vous vous êtes certainement posé la question .Pourquoi réécrire chaque fois ces séquences? Si vous vous êtes posé cette question vous êtes sur la bonne voie en programmation. En effet grâce aux sous-programmes, vous avez la possibilité d'écrire une seule séquence d'instructions et de l'appeler quand vous en avez besoin à différents emplacements de votre programme.

Un sous programme dans le langage informatique anglo-saxon s'appelle une "Sub Routine". Lorsque vous rencontrez au cours d'un programme l'instruction GOSUB, cela signifie "aller à la sous routine". De même lorsque le programme rencontre l'instruction RETURN, cela signifie "retourner au programme principal".

Nous allons apprendre comment utiliser les sous programmes dans des exemples simples.

Ces exemples vont nous permettre de comprendre la grande versatilité des sous-programmes.

4.2 La clarté d'un programme

Nous allons prendre 2 programmes Cf listing 4.1 et 4.2. Le premier programme est rédigé sans appel à des sous-programmes.Le deuxième programme fait appel à des sous programmes.

Examinons le programme 4.1:

Nous pouvons remarquer sans avoir fait de programmation que certaines instructions reviennent régulièrement dans le programme:

Les lignes 280 à 320 ainsi que les lignes 360 à390 sont identiques aux lignes 400 à 440/470 à 500 etc...

Maintenant,plongeons nous un peu plus dans la structure du programme et essayons d'analyser les tâches que le programme doit remplir.

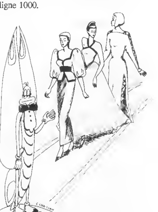
1°)Le programme doit proposer un menu de sélection .Qui dit menu de sélection implique test sur l'entrée clavier du choix.Cette partie de programme, lignes 100 à 270, est difficilement fractionnable.

2°)Le programme doit effectuer des opérations simples:
additions,soustractions,divisions,multi plications. Dans une opération simple, nous avons besoin de 2 nombres: N et M dans notre programme. Nous pouvons donc essayer de grouper les séquences d'instructions de l'introduction des nombres.En effet, nous repétons quatre fois la même opération.Cela reviendrait, en imageant un tant soit peu notre propos, à avoir 4 piles de papiers identiques sur son bureau que l'on prendrait à tour de rôle,(opération parfaitement absurde) lorsque l'on aurait besoin d'écrire.

Nous pouvons déjà grouper dans un sous programme l'introduction des nombres. Nous allons voir l'évolution de notre organigramme ainsi que le listing du programme chaque fois que nous faisons intervenir un sous programme.

CF organigramme 4/O/2 et listing 4.2.

L'organigramme 4/O/2 est moins chargé que le précédent. En effet l'introduction des valeurs passe par un sous-programme commun situé en ligne 1000.



4.3 APPEL D'UN SOUS PROGRAMME:

L'instruction qui appelle un sous programme est l'instruction GOSUB suivie du numéro de ligne où est logé le sous-programme.

La rencontre d'une instruction GOSUB n° de ligne provoque le saut du programme à la ligne spécifiée après le GOSUB. Les instructions du sous programme sont éxécutées jusqu'à la rencontre de l'instruction RETURN (Retour) qui renvoie le programme à la ligne suivant immédiatement la ligne d'appel du sous programme.Voir schéma S.4.1. Dans notre programme (listing 4.2), nous rencontrons 4 fois l'instruction GOSUB 1000. Nous trouvons ces instructions aux lignes 300,420,540,660.
Aux lignes 300,420,540,660 le programme se branchera à la ligne 1000 .
A la ligne 1040, l'instruction RETURN retournera le programme respectivement aux lignes 330,450,570,690.

Le sous-programme implanté en ligne 1000...1040 assurera l'introduction des valeurs.

Examinons maintenant notre listing 3.2.Il apparaît encore des répétitions de séquences identiques.
La première séquence évidente est la suivante:
CLS/CALL COLOR("0Wb").
Nous trouvons cette séquence aux lignes 280/290

400/410

520/530

640/650

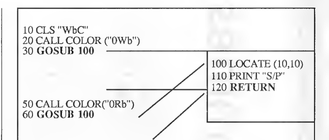
Il semble évident de construire un sous programme de déclaration des couleurs.
Problème :
Où implanter le sous-programme "couleur"?
Nous pouvons le placer soit après le sous programme INTRO soit avant.
Nous choisirons arbitrairement de loger ce programme à la ligne 1100.

Nouvelle étape:

Premièrement nous allons renuméroter notre programme grâce à la fonction RENUMBER.Nous choisirons arbitrairement une numérotation de 10 en 10 à partir de 100.Vous pourrez remarquer que la fonction RENUMBER renumérote tous les branchements.

RENUMBER 10,10 <ENVOI>

Il reste encore quelques séquences d'instructions que l'on peut grouper dans un sous programme.



70 END

**L'appel du sous-programme à la ligne 30 provoque un saut à la ligne 100. La ligne 120 renvoie le programme à la ligne qui suit la ligne d'appel du sous-programme.*

Ces séquences d'instructions se trouvent respectivement aux lignes 320 à 350, 410 à 440, 510 à 540, 600 à 640.
Nous pouvons donc élaborer un sous-programme que nous appellerons "affichage". Nous placerons ce sous programme à la ligne 800, immédiatement après le sous programme "couleur".

CF Listing 4.4 et l'organigramme O4.

Nous avons renuméroté le programme une fois le sous-programme affichage implanté, et nous avons supprimé les lignes inutiles. Par contre, nous avons rajouté quelques REM pour faire ressortir la structure du programme.

Examinons notre listing 4.4. Nous avons structuré notre programme de cette façon:

1° Partie du programme:
Ligne 100 à 270. Cette partie du programme affiche le menu de sélection, et ensuite, teste la valeur que nous donnons à la ligne 210. Les tests en cascade permettent de passer aux différentes parties du programme principal.

2° partie du programme:
Lignes 280 à 550. Toutes ces lignes gèrent les différentes opérations.
Nous pouvons étudier la partie "addition" de notre programme.

La ligne 310 affecte à la variable A$ la chaîne de caractères RESULTAT,à la variable B$ la chaîne de caractères "=", à la variable C$ la chaîne de caractères " +".
La ligne 320 calcule la valeur de R:Dans le cas de l'addition, R=N+M.
La ligne 330 appelle le sous-programme "affichage"(GOSUB 650) situé à la ligne 650. Le cycle de déroulement des instructions obéit aux mêmes lois que décrites plus haut:
Exécution des lignes du sous-programme jusqu'à une instruction RETURN-->retour au programme principal (ligne 340).
La ligne 340 nous renvoie au menu.
Nous avons vu le cycle de déroulement du programme. Nous pouvons répéter ce cycle pour les autres opérations.
En examinant attentivement le programme, nous pouvons nous apercevoir que les lignes 310, 380, 450, 520 se ressemblent beaucoup à une chaîne de caractères près. En effet la variable c$ contient les signes des opérations qui sont naturellement différents suivant l'opération choisie.
Nous pourrions essayer de grouper les 2 chaines de caractères communes au programme dans un sous-programme. La variable C$ serait seule présente dans les différentes parties du programme.
Une autre solution plus élégante serait d'écrire cette ligne:

-A$="RESULTAT:": B$="=" en amont du programme.En effet ,ces 2 affectations sont valables pour tout le programme. Nous pourrions placer cette ligne en 225 par exemple.

-La ligne 290 : GOSUB 610, appelle le sous-programme couleur situé à la ligne 610. Le programme passe à la ligne 610 où il effectue les instructions jusqu'à l'instruction RETURN(ligne 640). Une fois RETURN rencontré, l'ordinateur "passe la main" à l'instruction située immédiatement après l'appel du sous programme.(ligne 300). A ce niveau, on retrouve une autre instruction GOSUB. Le programme va donc retourner à un sous-programme(INTRO) situé à la ligne 560. Nous répétons l'opération décrite précédemment, à savoir:
Le programme effectue les lignes d'instructions du sous-programme.Une fois l'instruction RETURN trouvée, le programme est renvoyé à la ligne suivant l'appel du sous-programme(ligne 310).



Cette réduction est loin d'être finie. Si l'on se donne la peine de réfléchir et de penser programmation efficace, nous pouvons trouver certains cycles d'instructions qui sont valables pour le programme en entier.
En effet, le listing 4.1 est ce que l'on appelle communément dans le langage des programmeurs "un programme lourd". Le programme a été écrit sans avoir été pensé suffisamment.Nous avons certes organisé le programme en petites fractions plus courtes, mais il n'est pas structuré de manière convenable.

3.5 Structuration d'un programme

Essayons de voir ensemble les cycles d'instructions qui peuvent être mis en amont du choix. (avant les lignes 230 à 270) .Pour cela , regardons les lignes 290 à 300, 360 à 370, 430 à 440, 500 à 510. Les instructions qui les composent ne seront pas altérées par le choix de l'opération.
En effet, l'introduction des valeurs qui seront stockées dans les variables N et M ne seront pas modifiées par le choix de l'opérande.(signe +,-,/,X). Donc, ces lignes d'instructions ont leur place en amont du choix.
Seule la routine d'affichage donnera un résultat différent en fonction du choix de l'opérande.
Nous pouvons donc effectuer cette transformation dans notre programme.
Cf listing 4.5 / Organigramme O.5

Si nous comparons le programme initial 3.1 au programme final 3.5, nous pouvons nous rendre compte que le programme final 4.5 est beaucoup moins long et beaucoup plus clair en compréhension.
Nous pourrions économiser un peu plus de place en réduisant les lignes de tests. Pour cela, il suffirait de déclarer une instruction de branchement multidirections.

Nous reviendrons un peu plus loin sur cette notion. Nous avons annoncé en début de cette rubrique que les sous-programmes permettaient de réduire la place mémoire de votre programme.
Dans notre exemple le programme initial prenait 1840 octets, le programme final prend 1400 octets(et des brouettes). Nous pouvons donc dire que l'utilisation des sous programmes permet de réduire la taille mémoire.

3.6 Mise au point des programmes

La mise au point des programmes s'effectue sous-programmes par sous-programmes.Il est plus interressant de progresser de cette façon pour plusieurs raisons:

-1) Partage de la tâche:

-Avec l'utilisation des sous-programmes, on peut partager le travail. Untel se chargera de l'affichage, un autre pourra s'occuper des entrées etc.

-2) Facilités dans le debugage

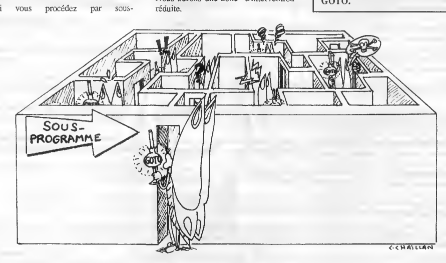
-Si vous procédez par sous-programmes, il est évident que lorsque vous aurez des corrections à apporter à votre programme, il sera plus facile d'aller "bidouiller" à la hauteur d'un sous-programme plutôt qu'au niveau du programme.

EX: Si nous vous demandons d'afficher le résultat en jaune plutôt qu'en rouge et de placer l'affichage en haut de l'écran, que faites vous? Si vous travaillez sur le programme initial, vous aurez à intervenir sur un grand nombre de lignes. Si, au contraire, vous faites la correction sur le listing 3.5, vous n'aurez qu'à transformer un petit nombre de lignes.
Nous aurons une zone "d'intervention" réduite.

-3° Vous pouvez modifier l'ordre l'appel des sous-programmes et ainsi modifier la séquentialité du programme.

-4° Si vous voulez rajouter de nouvelles intructions à votre programme, vous pouvez toujours créer un sous-programme plutôt que d'insérer des lignes entre 2 instructions. La liste de ces avantages n'est pas exhaustive.

Précaution d'emploi:
Il ne faut jamais essayer de rentrer dans un sous programme ni dans sortir à l'aide d'un GOTO.




```
1 0 '*********************************
110 '          LISTING 3.1           *
1 0 '*********************************
1 0 CLS
100 CALL COLOR("OWB")
1 0 LOCATE (  ):PRINT "1 ADDITION        "
140 LOCATE (11 ):PRINT "2 SOUSTRACTION    "
170 LOCATE (12 ):PRINT "3 DIVISION        "
160 LOCATE (13 ):PRINT "4 MULTIPLICATION  "
190 CALL COLOR("OWB")
200 LOCATE (17,12):PRINT "VOTRE CHOIX        "
210 LOCATE (15 21):ACCEPT BEEP  12  VALIDATE(DIGIT)NULL(  )
220 '
230 IF C=1 THEN GOTO 280
240 IF C=2 THEN GOTO 400
250 IF C=3 THEN GOTO 520
260 IF C=4 THEN GOTO 640
270 IF C=0 THEN GOTO 100
280 CLS
290 CALL COLOR("OWE")
300 LOCATE (10,1):PRINT "Donnez deux nombres:"
310 LOCATE (10,22):ACCEPT BEEP VALIDATE(NUMERIC)SIZE(5)NULL(0):N
320 LOCATE (10,30):ACCEPT BEEP VALIDATE(NUMERIC)SIZE(5)NULL(0):M
330 A$="RESULTAT:":B$="+":C$="="
340 R=N+M
350 CALL COLOR("ORH")
360 LOCATE (15,8):PRINT A$,N;C$;M;F$;R
370 LOCATE (16,8):PRINT A$;N;C$;M;F$;R
380 CALL COLOR("OWR"):LOCATE (18,8):PRINT "(TAPEZ RETOUR POUR MENU)":PAUSE
390 GOTO 100
400 CLS
```

```
410 CALL COLOR("OWB")
420 LOCATE (10,1):PRINT "Donnez deux nombres:"
430 LOCATE (10,22):ACCEPT BEEP VALIDATE(NUMERIC)SIZE(6)NULL(0):N
440 LOCATE (10,30):ACCEPT BEEP VALIDATE(NUMERIC)SIZE(6)NULL(0);M
450 A$="RESULTAT:":B$="=":C$=" "
460 R=N-M
470 CALL COLOR("ORH")
480 LOCATE (15,8):PRINT A$;N;C$;M;B$;R
490 LOCATE (16,8):PRINT A$;N;C$;M;B$;R
500 CALL COLOR("OWR"):LOCATE (19,8):PRINT "<TAPEZ RETOUR POUR MENU>":PAUSE
510 GOTO 100
520 CLS
530 CALL COLOR("OWF")
540 LOCATE (10,1):PRINT "Donnez deux nombres:"
550 LOCATE (10,22):ACCEPT BEEP VALIDATE(NUMERIC)SIZE(6)NULL(0):N
560 LOCATE (10,30):ACCEPT BEEP VALIDATE(NUMERIC)SIZE(6)NULL(0);M
570 A$="RESULTAT:":B$="=":C$="/"
580 R=INT(N/M)
590 CALL COLOR("ORH")
600 LOCATE (15,8):PRINT A$;N;C$:M:B$;R
610 LOCATE (16,8):PRINT A$;N;C$;M;B$;R
620 CALL COLOR("OWR"):LOCATE (19,8):PRINT "<TAPEZ RETOUR POUR MENU>":PAUSE
630 GOTO 100
640 CLS
650 CALL COLOR("OWB")
660 LOCATE (10,1):PRINT "Donnez deux nombres "
670 LOCATE (10,22):ACCEPT BEEP VALIDATE(NUMERIC)SIZE(6)NULL(0):N
680 LOCATE (10,30):ACCEPT BEEP VALIDATE(NUMERIC)SIZE(6)NULL(0);M
690 A$="RESULTAT:":B$="=":C$="X"
700 R=N*M
710 CALL COLOR("ORH")
720 LOCATE (15,8):PRINT A$;N;C$;M;B$;R
730 LOCATE (16,8):PRINT A$;N;C$;M;B$;R
740 CALL COLOR("OWR"):LOCATE (19,8):PRINT "<TAPEZ RETOUR POUR MENU>":PAUSE
750 GOTO 100
```

```
100 '**********************************
110 '*          LISTING 2             *
120 '**********************************
130 CLS "WBC"
140 CALL COLOR("OBW")
150 LOCATE (10,12):PRINT "1 ADDITION          "
160 LOCATE (11,12):PRINT "2 SOUSTRACTION      "
170 LOCATE (12,12):PRINT "3 DIVISION          "
180 LOCATE (13,12):PRINT "4 MULTIPLICATION    "
190 CALL COLOR("OWE")
200 LOCATE (15,12):PRINT "VOTRE CHOIX        ?"
210 LOCATE (15,31):ACCEPT BEEP SIZE(1)VALIDATE(DIGIT)NULL(0):C
220 '===========choix=================
230 IF C=1 THEN GOTO 280
240 IF C=2 THEN GOTO 400
250 IF C=3 THEN GOTO 520
260 IF C=4 THEN GOTO 640
270 IF C=0 THEN GOTO 100
280 CLS
290 CALL COLOR("OWB")
300 GOSUB 1000
330 A$="RESULTAT:":B$="=":C$="+"
340 R=N+M
350 CALL COLOR("ORH")
360 LOCATE (15,8):PRINT A$;N;C$;M;B$;R
370 LOCATE (16,8):PRINT A$;N,C$,M,B$ R
380 CALL COLOR("OWR"):LOCATE (19,8):PRINT "<TAPEZ RETOUR POUR MENU>":PAUSE
390 GOTO 100
400 CLS
410 CALL COLOR("OWB")
420 GOSUB 1000
450 A$="RESULTAT:":B$="=":C$="-"
460 R=N-M
470 CALL COLOR("ORH")
480 LOCATE (15,8):PRINT A$,N;C$;M;B$;R
490 LOCATE (16,8):PRINT A$;N,C$;M;B$;R
500 CALL COLOR("OWR"):LOCATE (19,8):PRINT "<TAPEZ RETOUR POUR MENU>":PAUSE
510 GOTO 100
520 CLS
530 CALL COLOR("OWB")
540 GOSUB 1000
570 A$="RESULTAT:":B$="=":C$="/"
580 R=INT(N/M)
590 CALL COLOR("ORH")
600 LOCATE (15,8):PRINT A$;N;C$;M;B$;R
610 LOCATE (16,8):PRINT A$;N;C$;M;B$;R
620 CALL COLOR("OWR"):LOCATE (19,8):PRINT "<TAPEZ RETOUR POUR MENU>":PAUSE
630 GOTO 100
640 CLS
650 CALL COLOR("OWB")
660 GOSUB 1000
690 A$="RESULTAT:":B$="=":C$="X"
700 R=N*M
710 CALL COLOR("ORH")
720 LOCATE (15,8):PRINT A$;N;C$;M;B$;R
730 LOCATE (16,8):PRINT A$;N;C$;M;B$;R
740 CALL COLOR("OWR"):LOCATE (19,8):PRINT "<TAPEZ RETOUR POUR MENU>":PAUSE
750 GOTO 100
1000 '********S/PROGRAMME INTRO*********
```

```
1010 LOCATE (10,1):PRINT "Donnez deux nombres:"
1020 LOCATE (10,22):ACCEPT BEEP VALIDATE(NUMERIC)SIZE(6)NULL(0);N
1030 LOCATE (10,30):ACCEPT BEEP VALIDATE(NUMERIC)SIZE(6)NULL(0);M
1040 RETURN
```

```
100 '**********************************
110 '*          LISTING 3.3            *
120 '**********************************
130 CLS "WbC"
140 CALL COLOR("ObW")
150 LOCATE (10,12):PRINT "1 ADDITION          "
160 LOCATE (11,12):PRINT "2 SOUSTRACTION      "
170 LOCATE (12,12):PRINT "3 DIVISION          "
180 LOCATE (13,12):PRINT "4 MULTIPLICATION    "
190 CALL COLOR("OWb")
200 LOCATE (15,12):PRINT "VOTRE CHOIX-   -   >"
210 LOCATE (15,31):ACCEPT BEEP SIZE(1)VALIDATE(DIGIT)NULL(0);C
220 '==============choix===============
230 IF C=1 THEN GOTO 280
240 IF C=2 THEN GOTO 400
250 IF C=3 THEN GOTO 520
260 IF C=4 THEN GOTO 640
270 IF C=0 THEN GOTO 100
280 GOSUB 1100
300 GOSUB 1000
330 A$="RESULTAT:":B$="=":C$="+"
340 R=N+M
350 CALL COLOR("ORH")
360 LOCATE (15,8):PRINT A$;N;C$;M;B$;R
370 LOCATE (16,8):PRINT A$;N;C$;M;B$;R
380 CALL COLOR("OWR"):LOCATE (19,8):PRINT "(TAPEZ RETOUR POUR MENU)":PAUSE
390 GOTO 100
400 GOSUB 1100
420 GOSUB 1000
450 A$="RESULTAT:":B$="=":C$=" "
460 R=N-M
470 CALL COLOR("ORH")
480 LOCATE (15,8):PRINT A$;N;C$;M;B$;R
490 LOCATE (16,8):PRINT A$;N;C$;M;B$;R
500 CALL COLOR("OWR"):LOCATE (19,8):PRINT "(TAPEZ RETOUR POUR MENU)":PAUSE
510 GOTO 100
520 GOSUB 1100
540 GOSUB 1000
570 A$="RESULTAT:":B$="=":C$="/"
580 R=INT(N/M)
590 CALL COLOR("ORH")
600 LOCATE (15,8):PRINT A$;N;C$;M;B$;R
610 LOCATE (16,8):PRINT A$;N;C$;M;B$;R
620 CALL COLOR("OWR"):LOCATE (19,8):PRINT "(TAPEZ RETOUR POUR MENU)":PAUSE
630 GOTO 100
640 GOSUB 1100
650 !
660 GOSUB 1000
690 A$="RESULTAT:":B$="=":C$="X"
700 R=N*M
710 CALL COLOR("ORH")
720 LOCATE (15,8):PRINT A$;N;C$;M;B$;R
730 LOCATE (16,8):PRINT A$;N;C$;M;B$;R
740 CALL COLOR("OWR"):LOCATE (19,8):PRINT "(TAPEZ RETOUR POUR MENU)":PAUSE
750 GOTO 100
1000 '********S/PROGRAMME INTRO*********
1010 LOCATE (10,1):PRINT "Donnez deux nombres:"
1020 LOCATE (10,22):ACCEPT BEEP VALIDATE(NUMERIC)SIZE(6)NULL(0);N
1030 LOCATE (10 30):ACCEPT BEEP VALIDATE(NUMERIC)SIZE(6)NULL(0);M
1040 RETURN
1100 '*********S/P COULEUR*************
1110 CLS "WbC"
1120 CALL COLOR("OWb")
1130 RETURN
```

```
100 '**********************************
110 '*          LISTING 3.4            *
120 '**********************************
130 CLS "WbC"
140 CALL COLOR("ObW")
150 LOCATE (10 12):PRINT "1 ADDITION          "
160 LOCATE (11 12):PRINT "2 SOUSTRACTION      "
170 LOCATE (12 12):PRINT "3 DIVISION          "
180 LOCATE (13,12):PRINT "4 MULTIPLICATION    "
190 CALL COLOR("OWb")
200 LOCATE (15,12):PRINT "VOTRE CHOI          "
210 LOCATE (15 31):ACCEPT BEEP SIZE(1)VALIDATE(DIGIT)NULL(0);C
220 '==============choix===============
230 IF C=1 THEN GOTO 290
240 IF C=2 THEN GOTO 360
250 IF C=3 THEN GOTO 420
260 IF C=4 THEN GOTO 490
270 IF C=0 THEN GOTO 100
280 '=============addition=============
290 GOSUB 610'APPEL SP COULEUR
300 GOSUB 560'APPEL SP INTRO
310 A$="RESULTAT:":B$="=":C$="+"
320 R=N+M
330 GOSUB 650'APPEL SP AFFICHAGE
```

```
340 GOTO 100
350 '=======soustraction========
360 GOSUB 610'APPEL SP COULEUR
370 GOSUB 560'APPEL SP INTRO
380 A$="RESULTAT:":B$="=":C$="-"
390 R=N-M
400 GOSUB 650'APPEL SP AFFICHAGE
410 GOTO 100
420 '=========division==========
430 GOSUB 610'APPEL SP COULEUR
440 GOSUB 560'APPEL SP INTRO
450 A$="RESULTAT:":B$="=":C$="/"
460 R=INT(N/M)
470 GOSUB 650'APPEL SP AFFICHAGE
480 GOTO 100
490 '========multiplication=======
500 GOSUB 610'APPEL SP COULEUR
510 GOSUB 560'APPEL SP INTRO
520 A$="RESULTAT:":B$="=":C$="X"
530 R=N*M
540 GOSUB 650'APPEL SP AFFICHAGE
550 GOTO 100
560 '********S/P INTRO****************
570 LOCATE (10,1):PRINT "Donnez 2 nombres"
580 LOCATE (10,22):ACCEPT BEEP VALIDATE(NUMERIC)SIZE(6)NULL(0);N
590 LOCATE (10,30):ACCEPT BEEP VALIDATE(NUMERIC)SIZE(6)NULL(0);M
600 RETURN
610 '********S/P COULEUR***************
620 CLS "WBC"
630 CALL COLOR("OWB")
640 RETURN
650 '********S/P AFFICHAGE*************
660 CALL COLOR("ORH")
670 LOCATE (15,8):PRINT A$;N;C$;M;B$;R
680 LOCATE (16,8):PRINT A$;N;C$;M;B$;R
690 CALL COLOR("OWR"):LOCATE (18,8):PRINT "<TAPEZ RETURN POUR MENU>":PAUSE
700 RETURN


100 '*******************************************
110 '*          LISTING 3,5                    *
120 '*******************************************
130 CLS "WBC"
140 CALL COLOR("OBW")
150 LOCATE (10,12):PRINT "1 ADDITION          "
160 LOCATE (11,12):PRINT "2 SOUSTRACTION      "
170 LOCATE (12,12):PRINT "3 DIVISION          "
180 LOCATE (13,12):PRINT "4 MULTIPLICATION    "
190 CALL COLOR("OWB")
200 LOCATE (15,12):PRINT "VOTRE CHOIX   -   >"
210 LOCATE (15,31):ACCEPT BEEP SIZE(1)VALIDATE(DIGIT)NULL(0),C
220 '=========choix===============
230 A$="RESULTAT:":B$="="
240 GOSUB 560'APPEL SP COULEUR
250 GOSUB 510'APPEL SP INTRO
260 IF C=1 THEN GOTO 310
270 IF C=2 THEN GOTO 360
280 IF C=3 THEN GOTO 410
290 IF C=4 THEN GOTO 460
300 IF C=0 THEN GOTO 100
310 '=========addition============
320 C$="+"
330 R=N+M
340 GOSUB 600'APPEL SP AFFICHAGE
350 GOTO 100
360 '=========soustraction========
370 C$="-"
380 R=N-M
390 GOSUB 600'APPEL SP AFFICHAGE
400 GOTO 100
410 '=========division============
420 C$="/"
430 R=INT(N/M)
440 GOSUB 600'APPEL SP AFFICHAGE
450 GOTO 100
460 '=========multiplication
470 C$="X"
480 R=N*M
490 GOSUB 600'APPEL SP AFFICHAGE
500 GOTO 100
510 '********S/P INTRO****************
520 LOCATE (10,1):PRINT "Donnez 2 nombres"
530 LOCATE (10,22):ACCEPT BEEP VALIDATE(NUMERIC)SIZE(6)NULL(0);N
540 LOCATE (10,30):ACCEPT BEEP VALIDATE(NUMERIC)SIZE(6)NULL(0);M
550 RETURN
560 '********S/P COULEUR***************
570 CLS "WBC"
580 CALL COLOR("OWB")
590 RETURN
600 '********S/P AFFICHAGE*************
610 CALL COLOR("ORH")
620 LOCATE (15,8):PRINT A$;N;C$;M;B$;R
630 LOCATE (16,8):PRINT A$;N;C$;M;B$;R
640 CALL COLOR("OWR"):LOCATE (18,8):PRINT "<TAPEZ RETURN POUR MENU>":PAUSE
650 RETURN
```

OFFRE SPECIALE

MICRO ORDINATEUR EXL 100
+ Moniteur monochrome vert.
+ Lecteur de cassettes.
+ Super valise logiciels comprenant :

- 2 logiciels cartouches ROM
- 4 cassettes d'initiation à l'EXELBASIC
- 2 cassettes "Faites parler votre EXL 100"
- 1 cassette "MILLE PATTES" jeu d'adresse
- 1 cassette "SUPER BAO" jeu de réflexion
- 1 cassette "CREAPHIX" logiciel graphic
- 3 cassettes vierges "Spécial informatique"
- 1 livre "25 programmes sur EXL 100"

= 3280 F TTC



EXL 100 + moniteur.



ROM.

Magnétophone.

Clavier professionnel en option 290 F.

Périphériques disponibles :

Clavier pro (mécanique) :	**290 F**
Paire de manettes de jeu :	**270 F**
Lecteur de cassette :	**290 F**
EXELMEMOIRE (16K RAM + 8K ROM) :	**550 F**
EXELMODEM : (fonctions MINITEL + repondeur automatique + télécommunications + interface imprimante + incrustation vidéo)	**1050 F**
EXELDRUMS : (boîte à rythmes professionnelle)	**1090 F**
Imprimante EXL 80 :	**3190 F**
Lecteur de disquettes EXL 135 : (1 Mega octets + 40 K RAM)	**3290 F**
Interface imprimante : (RSC 232 - Centronics - Incrustation vidéo)	**650 F**

1 AN DE GARANTIE TOTALE

* Dans la limite des stocks disponibles

Logiciels

ARCADE. Créez vos propres jeux d'Arcade	**169 F**
ATI 42. Simulateur de vol IFR :	**175 F**
BACK GAMMON. Un grand classique ! :	**179 F**
CITE D'OR. Fantastique jeu d'aventures :	**145 F**
L'ENIGME ALGENUBI. Un merveilleux jeu à énigmes :	**175 F**
EXELJACK. Jeu de Blackjack :	**99 F**
FIXEL. Gestionnaire de fichiers :	**120 F**
FITEXT. Traitement de texte :	**145 F**
GENEALOGIE. Traitement de fiches en arborescences :	**129 F**
INITIATION aux ADDITIONS :	**99 F**
INITIATION aux SOUSTRACTIONS :	**99 F**
REMI. Jeu musical et générateur de mélodies :	**129 F**
PUZZLE. Des puzzles de 6 à 700 pièces ! :	**149 F**
Verbes irréguliers allemands :	**99 F**
Verbes irréguliers anglais :	**99 F**
OPERAL. Mini-calculette parlante, répétiteur de tables :	**119 F**
TENNIS (cartouche) Un jeu de tennis exceptionnel :	**345 F**
IMAGIX (cartouche) Logiciel de création graphique :	**290 F**
WIZORO (cartouche) Jeu de combat spatial :	**290 F**
GUPPY (cartouche) Jeu de labyrinthe (un classique) :	**290 F**

LIVRAISON SOUS 48 H* en téléphonant au 93 73 64 64

BON DE COMMANDE à renvoyer à MICRO MAILING SERVICE 5, avenue Cyrille-Besset, 06800 Cagnes-sur-Mer

Nom ____________

Adresse ____________

____

Code postal ____________

Ville ____________

Règlement
☐ chèque bancaire/mandat/CCP ☐ contre-remboursement

Date Signature

ARTICLE	QUANTITE	PRIX TTC	TOTAL
TOTAL de la commande			

Participation aux frais de port de 0 à 500 F : 15 F — + de 500 F : 80 F

Frais de port : +

TOTAL A PAYER

Le catalogue des logiciels disponibles est expedie gratuitement avec la 1re commande

les astuces des programmeurs

Ce mois ci chers lecteurs et lectrices, Jean Luc Jonca vous propose deux programmes.

Le premier programme recopie sur imprimante EXL 80 l'écran graphique ou texte de votre ordinateur.

Le deuxième programme permet de convertir un programme assembleur en données BASIC. Ce programme est intéressant à plus d'un titre. Il vous permettra notamment de convertir les programmes assembleur proposés dans EV.

```
100 !            PROGRAMME DE RECOPIE D'ECRAN
110 DATA 142,196,156,119,255,22,15,181,114,20,31,213,29,232,
    142,197,155,226
120 DATA 4,114,255,22,10,34,28,142,197,174,138,194,173,45,24,230,9,178
130 DATA 136,120,80,26,208,27,224,57,192,138,194,157,56,0,114,24,27,74
140 DATA 1,27,230,5,142,196,254,224,22,210,27,138,193,3,192,138,193,2
150 DATA 88,82,41,0,152,1,26,142,197,220,142,196,254,138,194,172,192,138
160 DATA 194,171,152,1,26,138,194,173,178,208,27,142,197,220,34,29,142,197
170 DATA 174,213,22,10,138,194,168,208,19,138,
    194,157,176,190,208,21,138,194
180 DATA 170,208,23,138,194,169,208,22,114,40,20,142,197,145,152,23,25,82
190 DATA 5,200,152,25,14,246,142,197,83,201,171,0,25,120,122,25,121,0
200 DATA 24,202,236,142,197,117,120,3,23,121,0,22,218,20,219,34,53,142
210 DATA 197,174,120,234,23,121,1,22,218,21,199,34,53,142,197,174,10,128
220 DATA 40,128,36,119,4,19,1,180,184,128,40,128,36,119,2,19,1,180
230 DATA 184,128,40,128,36,119,1,19,1,180,201,100,201,100,10,114,8,16
240 DATA 114,2,14,82,5,170,0,25,191,223,14,171,0,25,202,245,18,14
250 DATA 142,197,174,218,16,232,10,82,128,34,64,142,197,174,202,249,10,166
260 DATA 4,4,13,34,2,139,1,50,138,1,49,35,13,45,9,10,179,10
270 DATA 184,34,2,139,1,50,185,139,1,49,34,3,139,1,50,34,1,139
280 DATA 1,49,34,3,139,1,49,34,2,139,1,50,138,1,49,38,16,250
290 DATA 138,1,49,35,29,45,9,230,247,10,213,19,213,22,152,26,14,124
300 DATA 82,19,72,1,14,73,0,13,152,14,24,246,213,20,142,197,145,114
310 DATA 40,21,128,40,128,36,145,40,145,36,83,127,38,16,5,39,8,2
320 DATA 35,240,208,17,193,230,3,116,128,22,136,5,0,16,39,8,13,38
330 DATA 16,6,136,0,0,16,224,4,136,10,0,16,72,1,16,121,0,15
340 DATA 82,10,152,16,14,200,246,201,128,40,128,36,171,196,79,120,128,16
350 DATA 121,0,15,202,235,118,16,17,10,119,1,17,6,116,4,22,142,199
360 DATA 133,142,198,187,142,198,255,82,8,170,196,89,36,64,184,142,197,174
370 DATA 185,119,2,22,3,142,197,174,202,237,119,2,22,10,115,253,22,210
380 DATA 21,120,4,24,224,3,120,2,24,121,0,23,152,24,14,246,218,21
390 DATA 2,224,3,140,197,248,115,239,22,34,53,142,197,174,118,8,22,16
400 DATA 116,8,22.224,7.77.27,19,227.5.211.19.140,197,224.10.115,247
410 DATA 22,119,4,22,10,117,1,22,118,1,22,3,115,251,22,224,226,114
420 DATA 8,16,213,15,82,1,119,8,22,2,82,6,170,196,79,184,35,127
430 DATA 190,171,196,79,185,35,128,200,188,93,6,231,2,90,5,202,247,201
440 DATA 68,0,15,195,119,8,22,4,93,11,231,220,93,6,231,216,50,16
450 DATA 210,16,18,15,188,171,196,89,93,1,230,192,10,18,17,192,35,7
460 DATA 87,16,2,208,20,18,20,83,224,206,206,206,118,16,17,44,119,4
470 DATA 17,19,152,1,16,82,8,170,196,89,180,35,254,171,196,89,202,245
480 DATA 152,16,1,118,128,22,15,119,2,17,3,116,2,22,119,1,17,3
490 DATA 116,4,22,10,34,7,115,239,22,184,206,170,199,174,208,15,170,199
500 DATA 175,208,16,201,206,170,199,174,208,13,170,199,175,208,
    14,82,8,154
510 DATA 16,208,18,154,14,208,17,170,196,89,67,0,18,180,35,254,19,17
520 DATA 20,18,171,196,89,211,16,121,0,15,211,14,121,0,13,202,220,115
530 DATA 127,22,10,82,5,118,1,22,10,170,196,79,171,196,84,202,248,224
540 DATA 8,170,196,84,171,196,79,202,248,82,10,204,
    170,196,79,206,171,196
550 DATA 79,194,171,196,79,202,242,10,199,246,199,238,199,230,
    199,222,199,214
560 DATA 199,206,199,198,199,190,0,0,0,0,0,0,0,0,16,0,36,0
570 DATA 16,0,36,0,34,8,18,0,34,8,18,0,4,16,58,16,4,16
580 DATA 58,16,42,20,42,20,42,20,42,20,24,50,24,50,24,50,24,50
590 DATA 46,54,46,54,46,54,46,54,255,255,255,255,255,255,255,255
600 FOR I=0 TO 879
610 READ A:CALL POKE(50318+I,A):NEXT
620 CALL EXEC(50318)
630 ! Ce programme permet d'imprimer un ecran sur l'imprimante EXL80.
640 ! Il sagit d'un programme assembleur contenant 880 octets.
650 ! Il doit etre charge en memoire avant toute utilisation.
660 ! Ce chargement est effectue par les lignes 600 et 610
670 ! L'appel du programme assembleur est obtenu par CALL EXEC(50318).
680 ! Lorsque l'ecran contient une zone graphique haute resolution,il est
690 ! necessaire de preciser au programme la couleur a ne pas imprimer.
700 ! Pour cela il suffit d'executer une commande CALL PLOT en specifiant
710 ! cette couleur comme parametre.
```




```
100 DIM A(20)
110 OPEN #1,"100.HRCOPY1",OUTPUT
120 FOR I=0 TO 880
130 CALL PEEK(50318+I,A(K))
140 K=K+1
150 IF K=18 THEN 250
160 NEXT I
170 LINE=LINE+10
180 PRINT #1,STR$(LINE)&" ";"DATA ";
182 PRINT STR$(LINE)&" ";"DATA ";
185 IF K=0 THEN 230
190 FOR R=0 TO K-1
200 PRINT #1,STR$(A(R));",";
210 NEXT R
220 PRINT #1,STR$(A(K))
230 CLOSE #1
240 STOP
250 K=0:LINE=LINE+10
260 PRINT #1,STR$(LINE)&" ";"DATA ";
265 PRINT STR$(LINE)&" ";"DATA ";
270 FOR R=0 TO 16
280 PRINT #1,STR$(A(R));",";
290 NEXT R
300 PRINT #1,STR$(A(17))
310 GOTO 160
385 IF K=0 THEN 230
```

TRESORS ET FANTOMES

de Christian Doussot

```
2 REM ______ C. DOUSSOT ______
4 CALL POKE(50432,162,5,45,162,136,45,10,162,5,45,162,200,45,10)
6 CALL EXEC(50432)
8 CALL POKE(50688,165,8,6,10):CALL POKE(49156,198,0)
10 CLS "CBb":RANDOMIZE
12 PR$=" TRESORS ET FANTOMES "
14 VIE=5:CALL CHATEAU
16 LOCATE (2,7):CALL COLOR("0MH"):PRINT PR$
18 LOCATE (3,7):CALL COLOR("0YH"):PRINT PR$
20 LOCATE (16,5):CALL COLOR("0GF"):PRINT "Appuyez sur une touche ..."
22 CALL MUSIPRES:LOCATE (16,5):PRINT RPT$(" ",26)
24 LOCATE (16,5):CALL COLOR("0RB"):PRINT "1/ CLAVIER   2/ MANETTE."
26 IF KEY$="1"THEN CLAV=1
28 LOCATE (18,5):CALL COLOR("0YB"):PRINT "Difficulte (1-5) ? ";
30 K$=KEY$:IF NUMERIC(K$)=0 THEN 30
32 IF VAL(K$)<1 OR VAL(K$)>5 THEN 30
34 DIF=VAL(K$)*3
36C$="077C312DD46D6F708213EC60072B58C10C66308211F4A0072D6841823E9316EAB63
7BAD"
38CC$="18D6E74A2139DE8443BDAD18E7634A319CD6846339AD18C6634A2118D68442D6
A518"
40 C1$="B5AD4A21AD5A84635AA518D4A54A21295A844294AD1FF"
42 MUSIQUE$=C$ëCC$ëC1$
44 CALL SPEECH("L,"ëMUSIQUE$)
46 PAUSE 1:DIF=DIF-1:IF DIF<2 THEN DIF=2
48 DIM A(30,30)
50 ATTACH TRESOR,CHOIX
52 CALL CHAR(65,"FFFFFFFFFFFFFFFFFF00")
54 CALL CHAR(66,"18183C5A991824242442")
56 CALL CHAR(67,"003F4385F98989898AFC")
58 CALL CHAR(79,"00000000001C38EE7700")
60 CALL CHAR(70,"00181818183C7E7E7E3C")
62 CALL CHAR(68,"03070F1F3F3F7F7FFFFF")
64 CALL CHAR(69,"C0E0F0F8F8FCFEFEFFFF")
66 CALL CHAR(71,"01010303030307070707")
68 CALL CHAR(72,"8080C0C0C0C0E0E0E0E0")
70 CALL CHAR(73,"070F0F0F0F1F1F1F1F3F")
72 CALL CHAR(74,"E0F0F0F0F0F8F8F8F8FC")
74 CALL CHAR(75,"3F3F3F3F3F7F7F7F7F7F")
76 CALL CHAR(76,"FCFCFCFCFCFEFEFEFEFE")
78 CALL CHAR(77,"7F7F7E7CF8F0F0E0C000")
80 CALL CHAR(78,"FEFE7E3E1F0F0F070300")
82 CALL CHAR(80,"FFFF7E3C3C1818000000")
84 CALL CHAR(81,"FFFFE7C3C3C3E7FFFFFF")
86 CALL CHAR(82,"FFFFFFFFFFFFFFFFFFFF")
88 CALL CHAR(83,"000000000018183C3C7E")
90 CALL CHAR(84,"0081C3E7EFFFFFFFFFFF")
92 CALL CHAR(85,"18101818081818101818")
94 CALL CHAR(86,"00000000FFFF00000000")
96 CLS "CBB":FOR I=8 TO 13:FOR J=13 TO 29:A(I,J)=0:NEXT:NEXT
98 CALL COLOR("0GB"):LOCATE (6,13):PRINT "POINTS DE VIE :";VIE
100 CALL CHATEAU
102 CALL COLOR("0CB"):LOCATE (15,15):PRINT "SCORE :";S
```



```
104 FO=INTRND(5)+10
106 FOR I=1 TO FO
108 CALL CHOIX(S,DIF,VIE,CLAV)
110 NEXT:CALL BONUS(VIE,S,MUSIQUE$)
112 AFF=0
114 CLS:CALL COLOR("1CB"):RESTORE 214
116 FOR Z=7 TO 14
118 READ X$
120 LOCATE (Z,11):PRINT X$
122 NEXT
124 FOR Z=11 TO 30:A(7,Z)=1:NEXT
126 FOR Z=11 TO 30:A(14,Z)=1:NEXT
128 FOR Z=7 TO 14:A(Z,11)=1:NEXT
130 FOR Z=7 TO 14:A(Z,30)=1:NEXT
132 LOCATE (8,29):CALL COLOR("1bB"):PRINT CHR$(5)
134 A(8,29)=5:A(13,12)=6
136 RESTORE 230
138 READ X,Y:IF X=99 THEN 142
140 A(X,Y)=1:GOTO 138
142 FOR X=8 TO 11:A(X,23)=1:NEXT
144 FOR X=9 TO 13:A(X,25)=1:NEXT
146 FOR X=9 TO 11:A(X,17)=1:NEXT
148 FOR X=8 TO 10:A(X,19)=1:NEXT
150 IF INTRND(2)=1 THEN 158
152 X=INTRND(6)+7:Y=INTRND(17)+12
154 IF Y/2<>INT(Y/2)THEN 152
156 A(X,Y)=7
158 FOR W=1 TO 2:RESTORE 246
160 FOR TT=1 TO 3
162 X=INTRND(6)+7:Y=INTRND(17)+12:IF Y/2<>INT(Y/2)THEN 162
164 LOCATE (X,Y)
166 READ TR$,CO$,NU
168 CALL COLOR("1"&CO$&"B"):PRINT TR$
170 A(X,Y)=NU
172 NEXT:NEXT
174 X=13:Y=12
176 CALL COLOR("0CB"):LOCATE (15,15):PRINT "SCORE :";S
178 CALL COLOR("0GB"):LOCATE (6,13):PRINT "POINTS DE VIE :";VIE
180 CALL COLOR("1MB")
182 CALL KEY1(C,B)
184 IF C=128 THEN M=1:P=0
186 IF C=129 THEN F=1:M=0
188 IF C=130 THEN M=-1:P=0
190 IF C=131 THEN P=-1:M=0
192 LOCATE (X,Y):PRINT " "
194 X=X-M:Y=Y+P:IF A(X,Y)=1 THEN X=X+M:Y=Y-P
196 LOCATE (X,Y):PRINT "B"
198 IF A(X,Y)=2 THEN CALL TRESOR(40,S,MUSIQUE$):A(X,Y)=0
200 IF A(X,Y)=3 THEN CALL TRESOR(30,S,MUSIQUE$):A(X,Y)=0
202 IF A(X,Y)=4 THEN CALL TRESOR(50,S,MUSIQUE$):A(X,Y)=0
204 IF A(X,Y)=5 THEN CALL TRESOR(200,S,MUSIQUE$):A(X,Y)=0:AFF=1
206 IF A(X,Y)=6 AND AFF=1 THEN CALL BONUS(VIE,S,MUSIQUE$):GOTO 36
208 IF A(X,Y)=7 THEN CALL PASSAGE(S,VIE,MUSIQUE$,DIF,C$):GOTO 96
210 IF INTRND(30)=1 THEN CALL CHOIX(S,DIF,VIE,CLAV)
212 GOTO 182
214 DATA AAAAAAAAAAAAAAAAAAAA
216 DATA A  A  A A A  A  A
218 DATA A A A A A A A A A AA
220 DATA A A  A A  A A   A
222 DATA A  A A  A A A A AA
224 DATA A A A  A A  A A  A
226 DATA A A  A A   A   A
```

```
228 DATA AVAAAAAAAAAAAAAAAAAAA
230 DATA 9,13,10,13,12,13,13,13
232 DATA 8,15,9,15,11,15,12,15
234 DATA 13,17
236 DATA 12,19,13,19
238 DATA 8,21,9,21,11,21,12,21
240 DATA 8,27,9,27,11,27,12,27
242 DATA 9,29,11,29
244 DATA 99,0
246 DATA C,R,2,F,G,3,O,Y,4
248 SUB TRESOR(BO,S,MUSIQUE$)
250 CALL SPEECH("L,"&MUSIQUE$)
252 S=S+BO:CALL COLOR("0CB"):LOCATE (15,15):PRINT "SCORE :";S
254 CALL COLOR("1MB"):SUBEND
256 SUB CHOIX(S,DIF,VIE,CLAV)
258 M$(1)=" DE":M$(2)=" GQQH":M$(3)=" IRRJ":M$(4)=" KRRL":M$(5)=" MPPN"
260 IF CLAV=1 THEN M=INTRND(4) ELSE M=INTRND(8)
262 T=0:CALL SPEECH("L,C78BAABAB602FC")
264 IF CLAV=1 THEN 270
266 ON M GOSUB 290,294,298,302,306,310,314,318
268 GOTO 272
270 ON M GOSUB 290,298,306,314
272 CALL COLOR("1"&CO$&"B")
274 FOR I=1 TO 5:LOCATE (X(I),Y(I)):PRINT (M$(I)):NEXT
276 T=T+1:CALL KEY1(A,B):IF A=TF THEN 322
278 IF T<>DIF THEN 276
280 VIE=VIE-1
282 CALL COLOR("0GB"):LOCATE (6,13):PRINT "POINTS DE VIE :";VIE
284 CALL SPEECH("L,BAB6020D7C91B2BAB60FC"):PAUSE .5
286 IF VIE=0 THEN CALL FIN(S)
288 GOTO 326
290 FOR I=1 TO 5:Y(I)=17:X(I)=I:NEXT
292 CO$="W":TF=128:RETURN
294 FOR I=1 TO 5:Y(I)=35:X(I)=I:NEXT
296 CO$="R":TF=79:RETURN
298 FOR I=1 TO 5:Y(I)=35:X(I)=I+9:NEXT
300 CO$="Y":TF=129:RETURN
302 FOR I=1 TO 5:Y(I)=35:X(I)=I+16:NEXT
304 CO$="W":TF=72:RETURN
306 FOR I=1 TO 5:Y(I)=16:X(I)=I+16:NEXT
308 CO$="M":TF=130:RETURN
310 FOR I=1 TO 5:Y(I)=1:X(I)=I+16:NEXT
312 CO$="b":TF=60:RETURN
314 FOR I=1 TO 5:Y(I)=1:X(I)=I+9:NEXT
316 CO$="G":TF=131:RETURN
318 FOR I=1 TO 5:Y(I)=1:X(I)=I:NEXT
320 CO$="M":TF=84:RETURN
322 S=S+10:PAUSE DIF/100
324 CALL COLOR("0CB"):LOCATE (15,15):PRINT "SCORE :";S
326 CALL COLOR("1MB")
328 FOR I=1 TO 5:LOCATE (X(I),Y(I)):PRINT "     ":NEXT
330 SUBEND
332 SUB FIN(S):CALL COLOR("0BC")
334 LOCATE (17,7):CALL COLOR("0BY"):PRINT "ICI SE TERMINE VOTRE VIE ..."
336 LOCATE (19,14):PRINT "SCORE : ";S:PAUSE .5
338 READ A,B
340 IF A=999 THEN 350
342 CALL POKE(258,A,132)
344 FOR J=1 TO 20+B*100:NEXT J
346 CALL POKE(259,45)
348 FOR J=1 TO 30:NEXT:GOTO 338
350 CALL COLOR("0MB")
```

```
352 LOCATE (21,8):PRINT "Appuyez sur une touche ...":K$=KEY$:CLS:RUN
354 DATA 176,3,176,2,176,1,176,2,149,2,158,1,158,2,176,1,176,2
356 DATA 188,1,176,3,999,1
358 SUBEND
360 SUB BONUS(VIE,S,MUSIQUE$)
362 BONUS=VIE*100
364 S=S+BONUS:LOCATE (17,15):CALL COLOR("0RH"):PRINT "BONUS : ";BONUS
366 LOCATE (18,15):PRINT "BONUS : ";BONUS
368 CALL SPEECH("L,"&MUSIQUE$):PAUSE 2
370 SUBEND
372 SUB CHATEAU
374 CALL CHAR(122,"0000000000000000FFFF")
376 CALL CHAR(33,"000000000003070FFFFF")
378 CALL CHAR(121,"0000000080C0E0F0FFFF")
380 CALL CHAR(35,"03070F3FFFFFFFFFFFFF")
382 CALL CHAR(36,"1F1F1F1F1F1F1F7F7FFF")
384 CALL CHAR(37,"1F191919191919191F1F")
386 CALL CHAR(38,"1F1F1F1F1F1F1F1F1F1F")
388 CALL CHAR(39,"F7F7F7F0F7F7F7F7F7F7")
390 CALL CHAR(40,"F7F7F7F7F7F7F7F7FFFF")
392 CALL CHAR(41,"FFFFFF00FFC0DFDFDFDF")
394 CALL CHAR(42,"DFDFDFDFDFDFDFC0FFFF")
396 CALL CHAR(43,"B7B7B7B7B7B7B737FFFF")
398 CALL CHAR(44,"F7F5F505F535B5B7B7B7")
400 CALL CHAR(45,"D8D8D8D8D8D8D8D8D8D8")
402 CALL CHAR(46,"D8D8D8D8D8D8D8FCFEFF")
404 CALL CHAR(47,"80E0E0F0F0F8F8FEFFFF")
406 CALL CHAR(48,"E0FFE1E1F7F7F7F7F7F7")
408 CALL CHAR(49,"FFFFFFCFCFCFCFCFFFFF")
410 CALL CHAR(50,"868587878F9FBFA3A3F7")
412 CALL CHAR(51,"EEBECEEEF6FAFE0606FE")
414 CALL CHAR(52,"FEFEFEFEFEFEFEFEFEFE")
416 CALL CHAR(53,"CFCFFEFEE6E6E6E6E6FE")
418 CALL CHAR(54,"004040E0E0F0F8FCFCFE")
420 CALL CHAR(55,"0000000000010307070F")
422 CALL CHAR(56,"1F1F87C7E7E787878787")
424 CALL CHAR(57,"87878787878787878787")
426 CALL CHAR(58,"FFCFCFCFCFCFCFCFFFFF")
428 CALL CHAR(59,"FCFEFFFFCFCFFFFFFFFF")
430 CALL CHAR(60,"101F1F10103838387C7C")
432 CALL CHAR(61,"0001010707070F1F0707")
434 CALL CHAR(62,"73731F1F18181F1F1F1F")
436 CALL CHAR(63,"0F0F0F8D8D9D9DCFE7F7")
438 CALL CHAR(64,"000101010303070F3C3C")
440 CALL CHAR(120,"FFFFFFFFFFFFFFFFFFFF")
442 FOR XX=7 TO 13:READ C$
444 IF C$="FIN"THEN 464
446 LOCATE (XX,11):CALL COLOR("1WB"):PRINT C$
448 NEXT XX
450 DATA "        <76"
452 DATA "       =;85"
454 DATA "      à?:94"
456 DATA "      ë0123"
458 DATA "       %'),-"
460 DATA "      $(*+."
462 DATA zz!yzzz!£xxxxx/zzzzz
464 SUBEND
466 SUB PASSAGE(S,VIE,MUSI$,DIF,SON$):DIM D(20,40)
468 CLS:CALL COLOR("1CB")
470 FOR I=1 TO 21
472 READ L$:LOCATE (1,1):PRINT L$
474 NEXT
```

```
476 DATA RR  RRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRR  RRR
478 DATA RR   RRRRRRRRRRRRRRRRRRRRRRRRRRRRRRR  RRR
480 DATA R    RRRRRRRRRRRRRRRRRRRRRRRRRRRRR   RRR
482 DATA RR    RRRRRRRRRRRRRRRRRRRRRRRRRRRRR    R
484 DATA RRR  RRRRRRRRRRRRRRRRRRRRRRRRRRRRRRR   RR
486 DATA RR   RRRRRRRRRRRRRRRRRRRRRRRRRRRRRRR  RRR
488 DATA R    RRRRRRRRRR RRRRRRRRRRRRRRRRRR   RRR
490 DATA RR   RRRRRRRRR  RRRRRRRR RRRRRRRRR    RR
492 DATA RR    RRRRRRRRR  RRRRRRR RRRRRRRRR   RRR
494 DATA RR                        R
496 DATA RR                       RR
498 DATA R                        R
500 DATA RR                        R
502 DATA RR                       RR
504 DATA R                       RR
506 DATA R                      RRR
508 DATA RR                       RR
510 DATA R                       RR
512 DATA R      S      S  S  S   S    RRR
514 DATA RRRRRRTRRRRRRTRRRTRRRRRRRRRRRRRRRRTRRRRRR
516 DATA RRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRRR
518 CALL COLOR("1MB")
520 FOR I=1 TO 5:READ XX:LOCATE (10,XX):PRINT "P":NEXT
522 DATA 9,12,19,23,33
524 CALL COLOR("1YB"):FOR I=18 TO 1 STEP -1
526 LOCATE (I,36):PRINT "U":NEXT
528 CALL COLOR("1RB")
530 FOR I=1 TO 5:READ Z:D(19,Z)=1:LOCATE (19,Z):PRINT "S":NEXT
532 FOR I=1 TO 4:READ Z:D(20,Z)=1:NEXT
534 DATA 11,20,24,27,32,7,14,18,34
536 CALL COLOR("1MB")
538 FOR I=2 TO 18:LOCATE (I,4):PRINT "B":CALL POKE(258,200-I*3,132)
540 PAUSE .05:LOCATE (I,4):PRINT " ":NEXT:CALL POKE(259,45)
542 CALL SPEECH("L,"&MUSI$):LOCATE (19,4):PRINT "B"
544 Z=4:PAUSE 1
546 Z=Z+1
548 LOCATE (19,Z-1):PRINT " "
550 LOCATE (2,12):CALL COLOR("0GB"):PRINT "SCORE :";S
552 LOCATE (4,10):PRINT "POINTS DE VIE :";VIE
554 CALL COLOR("1MB"):LOCATE (19,Z):PRINT "B":PAUSE DIF/100
556 IF Z>=37 THEN 576
558 IF D(19,Z)=1 OR D(20,Z)=1 THEN CALL SPEECH("L,"&SON$):VIE=VIE-1
560 IF VIE=0 THEN CALL FIN(S)
562 CALL KEY1(A,B)
564 IF A<>32 THEN 546
566 CALL SPEECH("L,C78BAABAB6027C")
568 LOCATE (19,Z):PRINT " ":Z=Z+2
570 LOCATE (18,Z-1):PRINT "B"
572 PAUSE .05:LOCATE (18,Z-1):PRINT " "
574 S=S+15:GOTO 554
576 CALL SPEECH("L,"&MUSI$):PAUSE .5
578 CALL COLOR("1MB"):FOR I=19 TO 2 STEP -1
580 LOCATE (I,37):PRINT "B":CALL POKE(258,200-3*I,132):LOCATE (I,37)
582 PAUSE .05:PRINT " ":NEXT
584 CALL POKE(259,45):SUBEND
586 SUB MUSIPRES
588 READ A,B:IF A=999 THEN RESTORE:CALL POKE(259,45):PAUSE 1:GOTO 588
590 CALL KEY1(X,Y):IF X<>255 THEN 610
592 CALL POKE(258,A,132)
594 FOR I=1 TO 30+B*200:NEXT:CALL POKE(259,45):GOTO 588
596 DATA 176,1,133,.75,133,.5,133,1
598 DATA 118,1,105,.75,105,.5,105,1.5,133,.5,118,.5,105,.5
```

```
600 DATA 99,1,149,1,118,1,133,2,105,.5,88,.5,105,.5,79,1.5,88,.5
602 DATA 88,.5,99,.5,99,1.5,99,.5,99,.5,118,1,88,1.5,99,1,99,.5,105,.5,105,1
604 DATA 176,1,133,.75,133,.5,133,1,118,1,105,.75,105,.5,105,1.5,133,.5
606 DATA 118,.5,105,.5,99,1,141,1,118,1,133,2
608 DATA 999,0
610 CALL POKE(259,45)
612 SUBEND
```



initiation au langage machine

Ecriture et lecture dans le VDP

1.1 Codage d'un caractère

Dans le dernier numéro d'Exelement vôtre, nous avons commencé à étudier l'organisation du 3556 ou le VDP (Video Display Processor). On définissait la structure de l'écran. Pour mémoire, l'écriture d'un caractère est codée sur 2 octets.
Le premier octet définit le code attribut. Cet octet est représenté de cette manière:

b7	b6	b5	b4	b3	b2	b1	b0

b7 b6 b5 définissent la couleur du caractère

b2 b1 b0 définissent la couleur de fond

b4 et b3 définissent le générateur de caractères

Le deuxième octet définit le code du caractère. Cet octet est représenté de cette manière:

b7	b6	b5	b4	b3	b2	b1	b0

b6 à b0 contient le code du caractère soit 128 valeurs. b0 indique si le caractère doit être mis en mode clignoatant ou non.

1.2 Organisation de la RAM VDP et de la RAM CPU

Pour ceux qui connaissent le langage machine, l'EXL 100 possède des particularités. Il n'est pas possible d'adresser directement la RAM VDP. Il est obligatoire de passer par un programme assembleur qui doit être implanté en RAM CPU.

Cette RAM CPU commence en >C000 et finit en >C7FF soit 49152 à 51199. C'est dans cet espace que les programmes assembleurs seront logés. On utilise généralement l'adresse 50688.
Pourquoi 50688 soit C600 Hexa? Les 2 Ko de la RAM CPU sont organisées de la façon suivante:
De >C000 à >C300 soit 49152 à 49920, on trouve tous les pointeurs utilisés par le système et notamment l'adresse de début de l'écran texte en> C103 et >C102. Nous vous donnons en annexe les adresses de ces pointeurs.

A partir de >C300 sont stockés les mots clés utilisateurs. Il est donc préférable de charger vos données vers la fin de la RAM CPU. 50688 est une bonne adresse de stockage.
La mémoire VDP dans laquelle se trouve vos programmes BASIC est organisée de cette manière cf schéma 1. Cette mémoire n'est pas directement adressable avec des CALL PEEK et des CALL POKE. Nous allons voir ensemble comment écrire et lire le VDP en assembleur.
L'assembleur du TMS 7000 possède 3 instructions qui permettent la lecture et l'écriture du VDP.

1. **TRAP 8:** Positionnement du pointeur d'écriture.

2. **TRAP 9:** Positionnement des pointeurs de lecture et d'écriture à la même adresse.

3. **TRAP 10:** Positionnement des pointeurs d'écriture et lecture à des adresses différentes.

Ces pointeurs contiennent l'adresse où le VDP sera écrit ou lu. Un excellent exemple d'application consiste dans:

-1- Effacement de la ligne de contrôle

-2- Traitement de l'écran (modification de la couleur de fond et la couleur du caractères)

1.3 Effacement de la ligne de contrôle:

Si vous êtes lecteur d'Exelement vôtre, vous avez certainement essayé de taper le programme d'effacement de la ligne de contrôle. Nous vous donnons ici un autre exemple qui fait appel aux particularités d'un registre du VDP. Le VDP possède 16 registres de 8 bits. Parmi ces 16 registres, le registre CM2 qui est le sixième registre du VDP permet d'afficher ou non la ligne de contrôle, ainsi qu'une grille. Le petit programme assembleur permet d'effacer la ligne de contrôle .

Vous pouvez directement exploiter ce programme en traduisant les codes en décimaux et en les implantant grâce à des CALL POKE en 50688. Nous vous donnons la procédure nécessaire ci dessous:

-1°) A partir du listing source

assembleur, il faut extraire les codes hexadécimaux de la deuxième colonne:

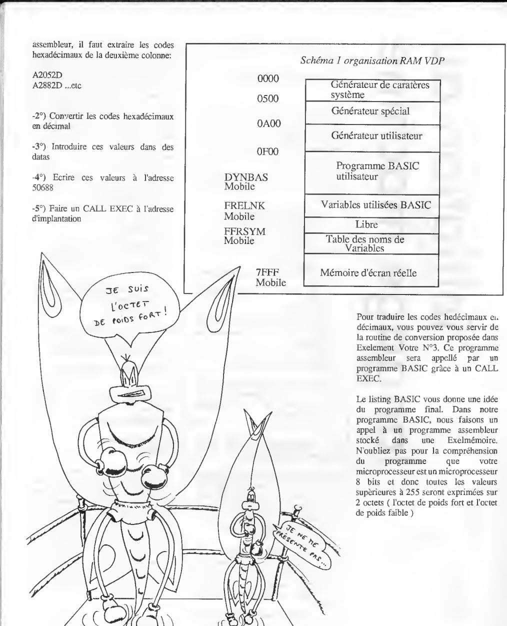
A2052D
A2882D ...etc

-2°) Convertir les codes hexadécimaux en décimal

-3°) Introduire ces valeurs dans des datas

-4°) Ecrire ces valeurs à l'adresse 50688

-5°) Faire un CALL EXEC à l'adresse d'implantation



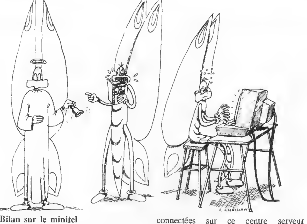
Pour traduire les codes hedécimaux en décimaux, vous pouvez vous servir de la routine de conversion proposée dans Exelement Votre N°3. Ce programme assembleur sera appellé par un programme BASIC grâce à un CALL EXEC.

Le listing BASIC vous donne une idée du programme final. Dans notre programme BASIC, nous faisons un appel à un programme assembleur stocké dans une Exelmémoire. N'oubliez pas pour la compréhension du programme que votre microprocesseur est un microprocesseur 8 bits et donc toutes les valeurs supèrieures à 255 seront exprimées sur 2 octets (l'octet de poids fort et l'octet de poids faible)

```
0000                ; MODIFICATION DE LA COULEUR DE LIGNES BASIC
0000                ; PARAMETRES: R124 nouvelle couleur du caractere
0000                ;             R125 premiere ligne
0000                ;             R126 nombre de lignes
0000                ;             R127 nouvelle couleur de fond
0000                ;
0000                ; suppression de la ligne de controle
0000 A20520                 MOVP    05,P45
0003 A2882D                 MOVP    088H,P45
0006                ; calcul de l'adresse ecran
0006 8877FE0E               MOVD    077FEH,R14
000A 7C527D                 MPY     82,R125
000D 48010E                 ADD     B,R14
0010 43000D                 ADC     A,R13
0013 F6                     TRAP    3
0014                ; alignement couleur caractere
0014 DE7C                   RL      R124
0016 DE7C                   RL      R124
0018 DE7C                   RL      R124
001A DE7C                   RL      R124
001C DE7C                   RL      R124
001E                ; initialisation du registre R20 a 40 (40 caracteres par lign
001E 722814         LOOP1   MOV     40,R20
0021                ; lecture d'un octet (ATTRIBUT)
0021 D728           LOOP2   DW      0D728H
0023                ; test generateur graphique
0023 270804                 BTJZ    008H,A,NEXT
0026                ; modification de la couleur de fond
0026 23F8                   AND     0F8H,A
0028 187F                   ADD     R127,A
002A                ; modification de la couleur du caractere
002A 231F           NEXT    AND     01FH,A
002C 147C                   OR      R124,A
002E                ; reecriture de l'attribut modifie
002E 822E                   MOVP    A,P46
0030                ; lecture d'un octet (CODE CARACTERE)
0030 D728                   DW      0D728H
0032                ; reecriture de cet octet
0032 822E                   MOVP    A,P46
0034                ; test fin de la boucle (40)
0034 DA14EA                 DJNZ    R20,LOOP2
0037                ; lecture et reecriture des 2 octets supplementaires par lign
0037 D728                   DW      0D728H
0039 822E                   MOVP    A,P46
003B D728                   DW      0D728H
003D 822E                   MOVP    A,P46
003F                ; test fin de boucle (nombre de lignes)
003F DA7EDC                 DJNZ    R126,LOOP1
0042                ; retour au programme BASIC
0042 0A                     RETS
0043 0000
```

```
100 ! MODIFICATION DE LA COULEUR DE LIGNES BASIC
110 CALL HROFF:CLS
120 CALL COLOR("0BC")
130 LOCATE (1,1) .
140 FOR I=1 TO 18
150 PRINT "LIGNE ";:PRINT I
160 NEXT
170 CALL COLOR("0BHL"):LOCATE (10,1)
180 PRINT "DDOOUUBBLLEE  HHAAUUTTEEUURR  &&"
190 PRINT "DDOOUUBBLLEE  HHAAUUTTEEUURR  &&"
200 PRINT "DDOOUUBBLLEE  LLAARRGGEEUURR"
210 PRINT "DDOOUUBBLLEE  LLAARRGGEEUURR"
220 CALL COLOR("0BC")
230 LOCATE (19,1)
240 PRINT "NUMERO DE LA PREMIERE LIGNE: ";
250 ACCEPT VALIDATE(DIGIT)SIZE(2),A
260 LOCATE (20,1)
270 PRINT "NOMBRE DE LIGNES A MODIFIER: ";
280 ACCEPT VALIDATE(DIGIT)SIZE(2),B
290 LOCATE (21,1)
300 PRINT "NOUVELLE COULEUR DE FOND (0 A 7):    ";
310 ACCEPT VALIDATE(DIGIT)SIZE(1),C
320 LOCATE (22,1)
330 PRINT "NOUVELLE COULEUR CARACT. (0 A 7):    ";
340 ACCEPT VALIDATE(DIGIT)SIZE(1),D
350 CALL POKE(124,D,A,B,C)
360 CALL DO("VDP")
```

télématique news



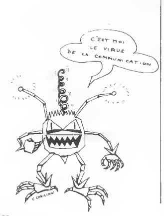
Bilan sur le minitel

Le nombre de postes minitel installés a dépassé depuis peu le million d'exemplaires. Malgré la tarification abusive de minitel, de plus en plus de personnes utilisent ses services. Récemment les pouvoirs publics ont du se rendre à l'évidence: Le nombre de lignes Transpac en service est insuffisant; et si l'on peut employer le néologisme, il faut faire la queue pour obtenir les services suivant l'heure d'appel. Malgré toutes ces péripéties, l'opération Minitel est un succès. Moi même, je l'avoue, j'y ai pris goût, et je "minitèle" plus que mon budget (tenu sur EXL 100) m'y autorise. Le virus de la communication s'est abattu sur moi comme la misère sur le tiers monde.

Une nouvelle culture est née

J'ose l'affirmer, une nouvelle forme de culture est en train de naître dans notre douce France. Les branchés de la télécom... sévissent sur les réseaux. Des dialogues se nouent entre personnes qui ne se connaissent pas, un courant d'amitié inonde les lignes; bref, les Français parlent aux français. Etonnant, à l'heure où tout le monde, pouvoir public en tête se plaint du manque de communication, à toutes les heures de la journée des dialogues improvisés se tissent par minitels interposés.La diversité dans le style de dialogue peut surprendre le néophyte qui se branche pour la première fois sur une messagerie directe.

Ah, mais il se peut que vous ne connaissiez pas les messageries directes. Nous allons remédier à ce grave manquement dans votre culture télématique. Une messagerie directe permet aux différentes personnes connectées sur ce centre serveur d'échanger des messages pratiquement en temps réel.



Expérience et premier contact

Lorsque je décidai d'expérimenter une messagerie, j'eus le trac. Qu'allait-il m'arriver? La gorge sèche et les paumes moites, je pianotai sur mon EXL 100, le numéro fatidique 16/3/615 91 77.

Le sifflement de la porteuse me fit brusquement prendre conscience que la liaison était établie. Je recherchai parmi les différents services, une messagerie. Brusquement j'apperçu sur l'écran: Aline messagerie en direct. Allez, encore un petit effort, du cran, que diable! Une fois la connection établie et les préliminaires d'entrée dans la messagerie finis (choix d'un pseudonyme etc...) , les noms des personnes branchées me fit siffler. OAUAH que de monde! A peine entré dans la danse, une personne m'envoya un message. Allez hop, je le lus et lui renvoyai le mien. Total une nuit blanche et une bonne note de téléphone. Je ne pourrais pas vous raconter tout ce qui s'est échangé comme messages cette nuit là. La seule chose à dire: Il y en a pour tous les goûts. De l'intello en quête de l'absolu en passant par du licencieux un peu démodé en faisant un détour du côté de la micro. Bref un monde hallucinant grouillant de vitalité. Emballé par l'expérience nocturne, je récidivai comme un collégien la journée suivante. Même densité de personnes branchées, même enthousiasme, sauf pour le portefeuille. Deviendrez vous des branchés du dialogue? Pourquoi pas. Il vous suffit d'acheter EXLMODEM.

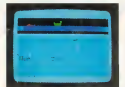
Les Editions PARALLELES vous proposent:

OPERAL Synthèse vocale
Sélectionné par l'Education Nationale, ce jeu permet d'apprendre les tables d'addition et de multiplication et d'initier l'enfant à l'utilisation des calculettes.
De 6 à 12 ans - 129 F.

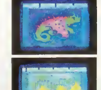


ARCADE Langage machine
Créez vos propres jeux d'ARCADE grâce aux utilitaires fournis. Gestion des lutins automatiques et manettes. Gestion des collisions, etc...
Livré avec un exemple de jeu réalisé grâce à ARCADE. 189 F.





NUMERIX Synthèse vocale
Des jeux pour apprendre à lire et à écrire les nombres (en chiffres et en lettres) et à les prononcer. L'EXL 100 peut prononcer tous les nombres jusqu'à 12 chiffres.
De 5 à 12 ans - 129 F.



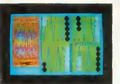
PUZZLE Langage machine
Jeu très performant. Permet de reconstituer à l'écran des graphismes découpés en un nombre de pièces variable allant de 8 à 640. Deux graphismes sont livrés avec la cassette. D'autres graphismes sont disponibles sur cassette séparée. 189 F.

PLURIEL Face 1
Jeu d'initiation au pluriel des noms et adjectifs. L'enfant apprend les pluriels tout en jouant au chat et à la souris.



LES PARAS
Vous êtes capitaine de navire et votre mission est de repêcher les parachutistes tombés du ciel, et de les ramener sains et saufs sur la rive. 149 F.

PLURIEL Face 2
Ce logiciel permet un enseignement complet des pluriels. A savoir, apprentissage des règles et exceptions pour tous les pluriels.
De 5 à 12 ans et plus - 149 F.



BACKAMMON
Un jeu de logique passionnant pour jouer à deux ou seul contre l'ordinateur. Plusieurs niveaux de jeux pour l'ordinateur.
189 F.

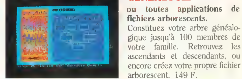
ALGOS
Ces jeux permettent à l'enfant de s'éveiller aux notions de rythmes. Trois jeux sont livrés: rythmes logiques, apprentissage des couleurs, rythmes musicaux.
De 4 à 12 ans - 129 F.



CRAPETTE
Un jeu de cartes original pour jouer à deux ou seul contre l'ordinateur.
149 F.



MUSICAL Synthèse vocale
Jeu très complet d'initiation au solfège. Permet d'apprendre à reconnaître le son, le dessin ainsi que le nom des notes. Une dictée musicale permet le contrôle des connaissances.
De 7 à 77 ans - 149 F.



GENEALOGIE
ou toutes applications de fichiers arborescents.
Constituez votre arbre généalogique jusqu'à 100 membres de votre famille. Retrouvez les ascendants et descendants, ou encore créez votre propre fichier arborescent. 149 F.

BON DE COMMANDE

Nom ____________________

Prénom ____________ Age ________

Adresse ____________________

Je désire recevoir

Logiciel	Prix	Logiciel	Prix

Total: ________

Ci-joint, mon règlement à l'ordre de Ed. PARALLELES - 143 bis route de l'Abadie -06730 SAINT ANDRE